

NAS8-35591

NASA CR-

(NASA-CR-16-1970) AUTOMATED LONGWALL
GUIDANCE AND CONTROL VERTICAL CONTROL
SUBSYSTEM, VOLUME 2 Final Report

(Foster-Miller Associates, Inc., Waltham,
Mass.) 204 p HC A18/MF A01

N82-18-59

Unclass

CSOI ORI 63/83 09157

AUTOMATED LONGWALL GUIDANCE AND CONTROL VERTICAL CONTROL SUBSYSTEM

VOLUME II

W.R. Griffiths, M. Smirlock, J. Aplin,
R. Fish, D. Fish
Foster-Miller Associates, Inc.

January 1982

Prepared for:

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
George C. Marshall Space Flight Center
Marshall Space Flight Center, AL



REPORT DOCUMENTATION PAGE

1. Report No. NASA CR-		2.		3. Recipient's Accession No.	
4. Title and Subtitle Automated Longwall Guidance and Control Vertical Control Subsystem				5. Report Date 31 January 1982	
				6.	
7. Author(s) W. Ronald Griffiths, Martin Smirlock, James Aplin, Roger E. Fish, David Fish				8. Performing Organization Report No. NAS7946	
9. Performing Organization Name and Address Foster-Miller Associates, Inc. 350 Second Avenue Waltham, MA 02254				10. Project/Task/Work Unit No.	
				11. Contract or Grant No. NAS8-33591	
12. Sponsoring Organization Name and Address National Aeronautics and Space Administration Geo. C. Marshall Space Flight Center Marshall Space Flight Center, AL 35812				13. Type of Report Contractor Report	
				14.	
15. Supplementary Notes					
16. Abstract A design fabrication and implementation of a horizon control of a longwall shearer has been performed. This equipment was tested and demonstrated aboveground at the Department of Energy's surface test facility in Bruceton, PA. This hardware was also installed on a longwall face in the Herrin No. 6 seam in southern Illinois. The feasibility of providing horizon control for a shearer was demonstrated aboveground. The feasibility of retrofitting the necessary sensors in a survivable manner was demonstrated underground. Subsequent tests of a specific component, the natural background sensor, at a western location demonstrated the particular usefulness of this device on a wider application basis.					
17. Originator's Key Words Horizon control Underground mining Longwall shearer control Coal interface detection				18. Availability Statement	
19. U.S. Security Classif. of the Report Unclassified	20. U.S. Security Classif. of This Page Unclassified		21. No. of Pages 573		22. Price

LIST OF APPENDICES

	Page
APPENDIX A - SOFTWARE DOCUMENTATION	151
APPENDIX B - PROGRAM LISTING	365

APPENDIX A
SOFTWARE DOCUMENTATION

· VERTICAL CONTROL SYSTEM

NAS-7946

SOFTWARE DOCUMENTATION

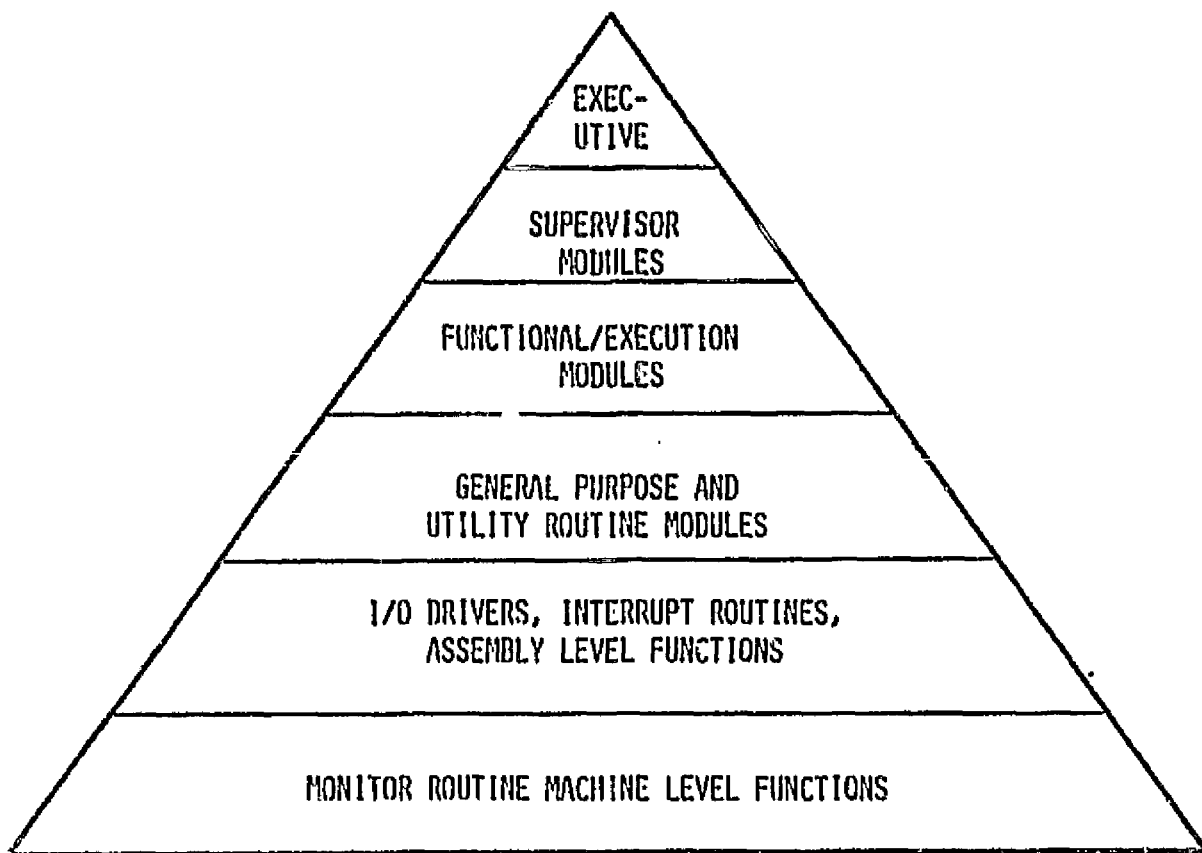
January 1982

Roger Fish

VERTICAL CONTROL SYSTEM

PROGRAM INTRODUCTION

VCS PROGRAM STRUCTURE



PROGRAM CAPABILITY

- SENSOR RANGE ERROR CHECKING
- MULTIPLE RUN MODES (CONTROL LAWS)
- MASK CONTROL OF I/O AND OPERATIONS
- NOT DYNAMICALLY SELF PROGRAMMING
- INTERACTIVE INITIALIZATION
- INTERACTIVE DEBUG OF I/O, MEMORY

OPERATIONAL FEATURES

DUAL Mode:

- RUN MODE (AUTOMATIC)
- DEBUG MODE (INTERACTIVE)

SYSTEM INITIALIZATION:

- PARAMETER SET FROM RAM
 - CLEAR RAM BUFFERS
 - SET UP POINTERS AND DEFAULT FLAGS
 - OPERATOR INTERACTION
- MENU BASED MODIFICATIONS PROMPT FOR CLASS AND
ELEMENT CHANGE FLAGS, VARIABLES

OPERATIONAL FEATURES (CONTINUED)

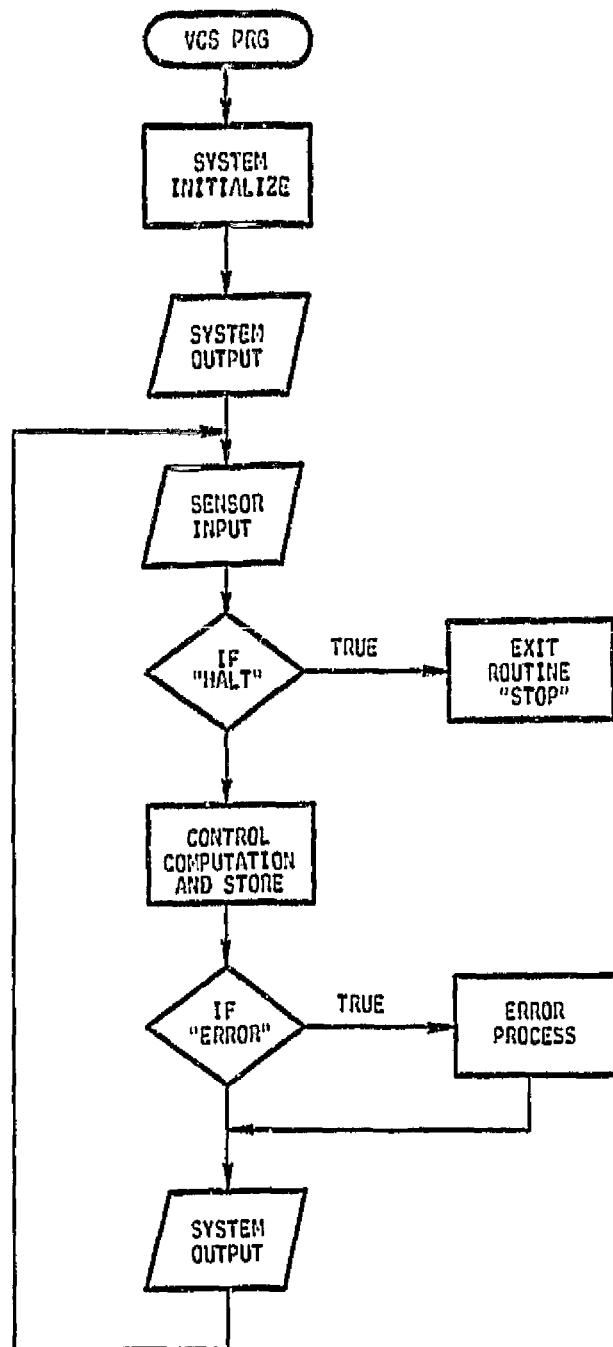
RUN:

- INPUT SENSORS (RANGE CHECK)
- CONTROL FUNCTION COMPUTATION
- OUTPUT CONTROL/DATA
- ERROR TEST
- SYNCHRONIZE TO CLOCK

DEBUG:

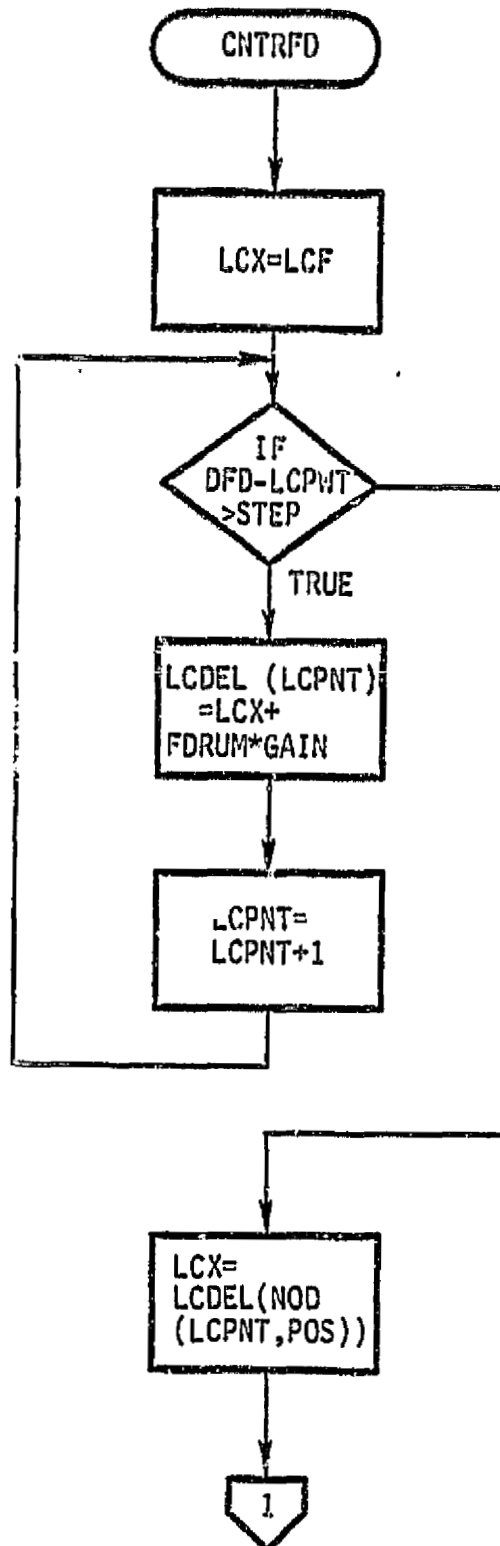
- INTERACTIVE SENSOR TEST
 - OUTPUT TEST
 - MEMORY CHECK/DUMP
 - RESTART/INITIALIZE

NORMAL VCS
PROGRAM FLOW

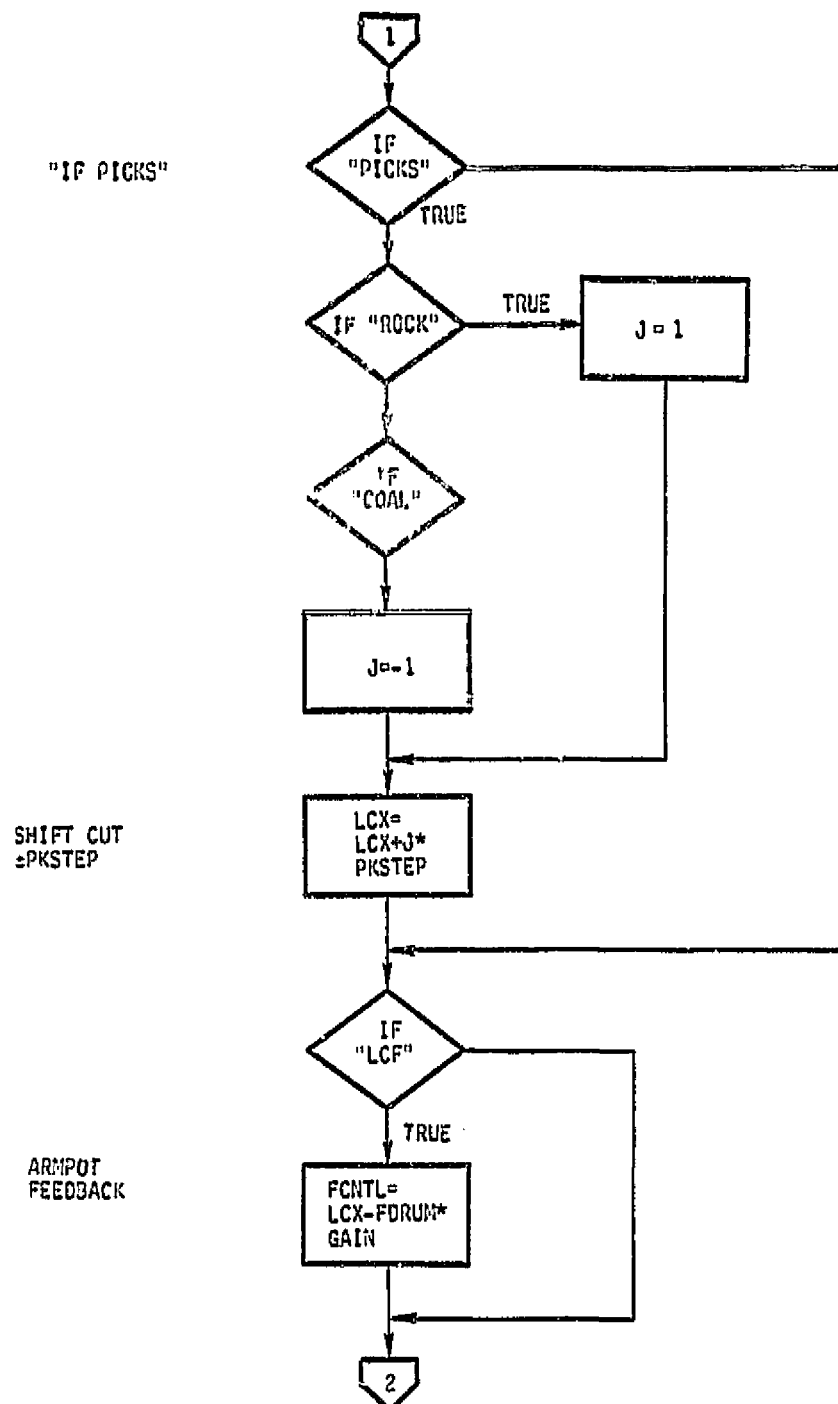


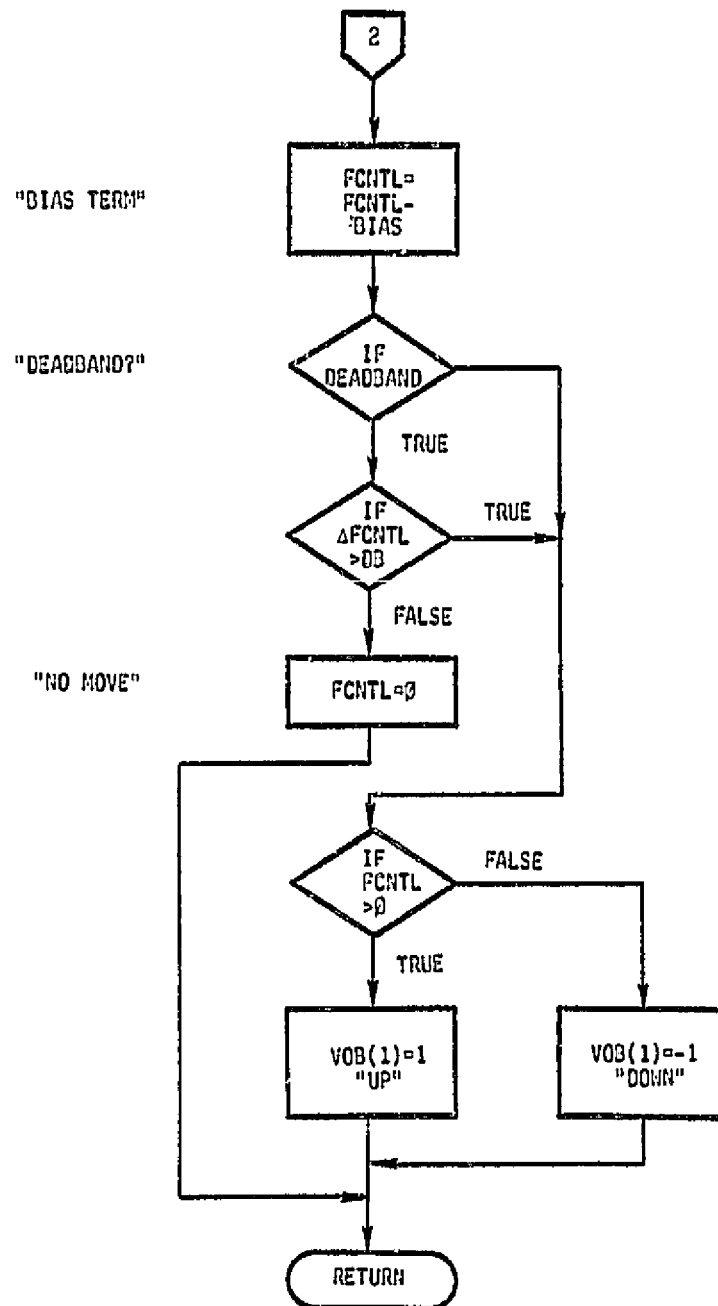
VCS OLD BEN CONTROL ALGORITHM "MODE II"

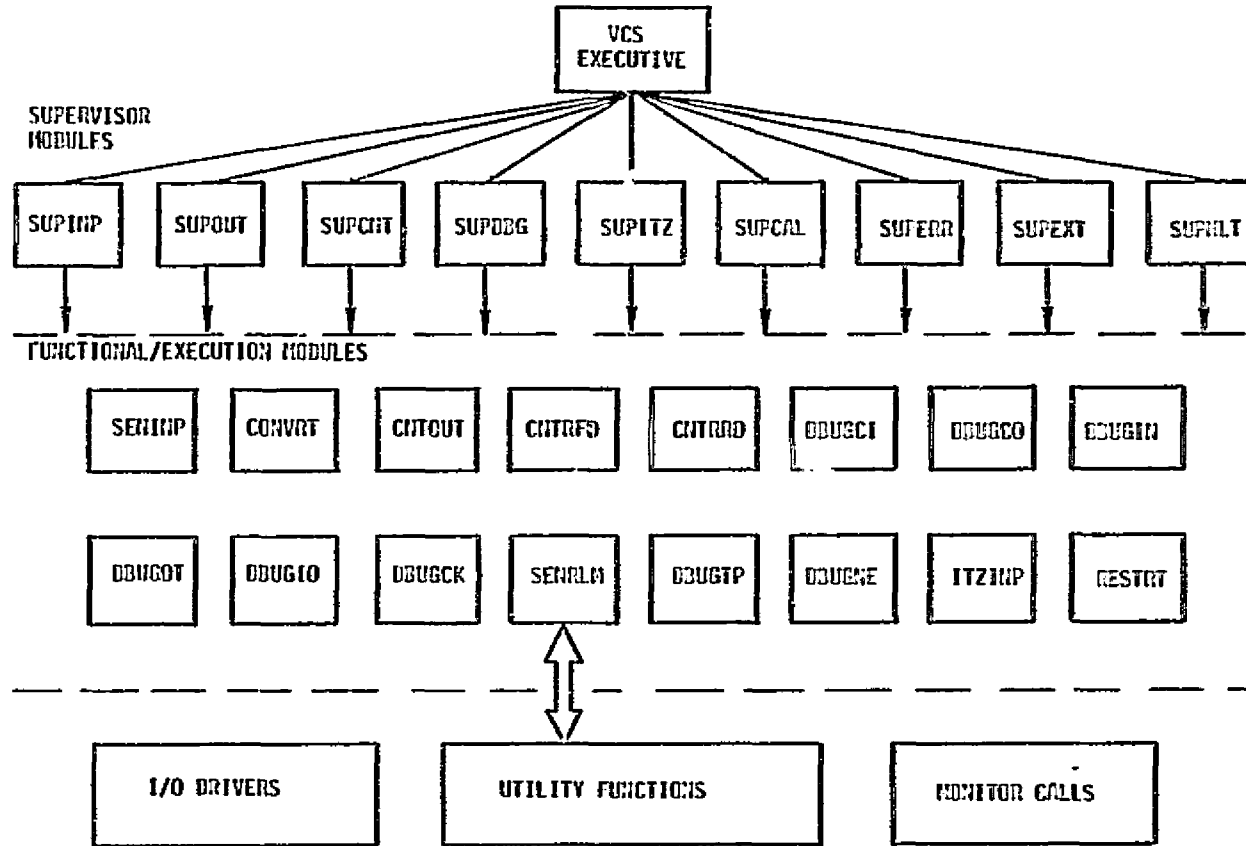
DELAYED SENSORS
NO PICK AVERAGE

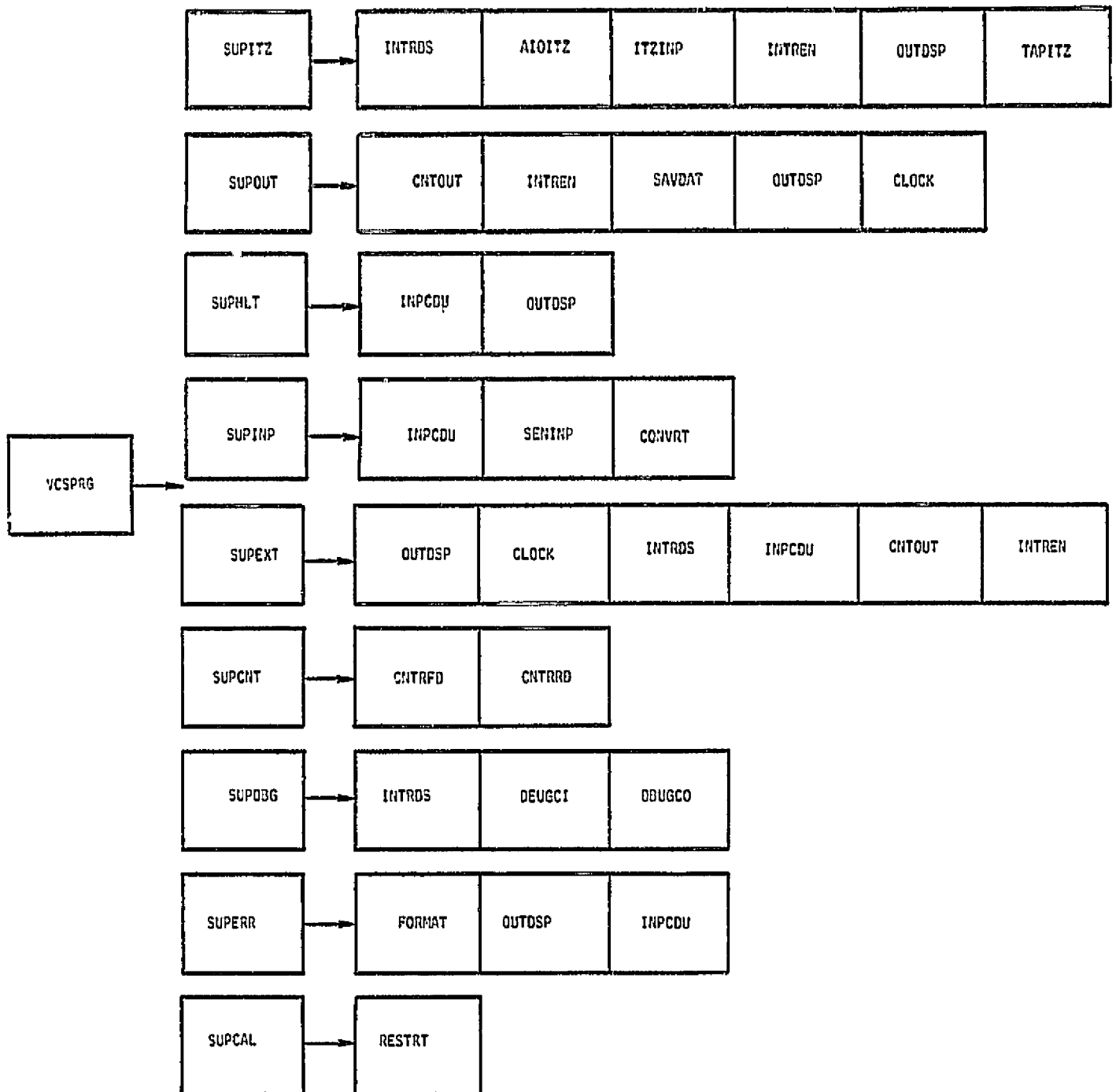


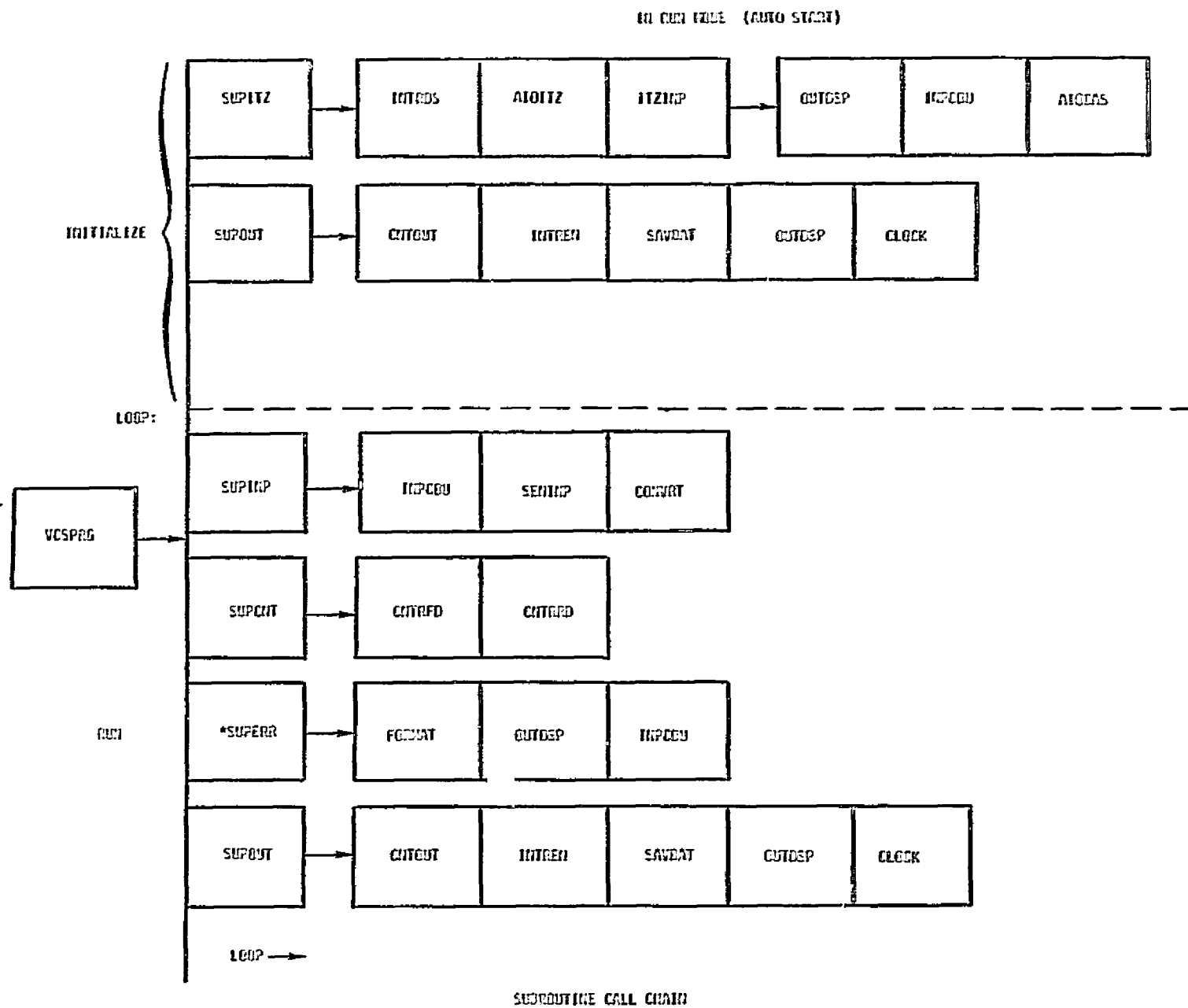
LCX = DELAYED
(LCF + FDRUM * GAIN;

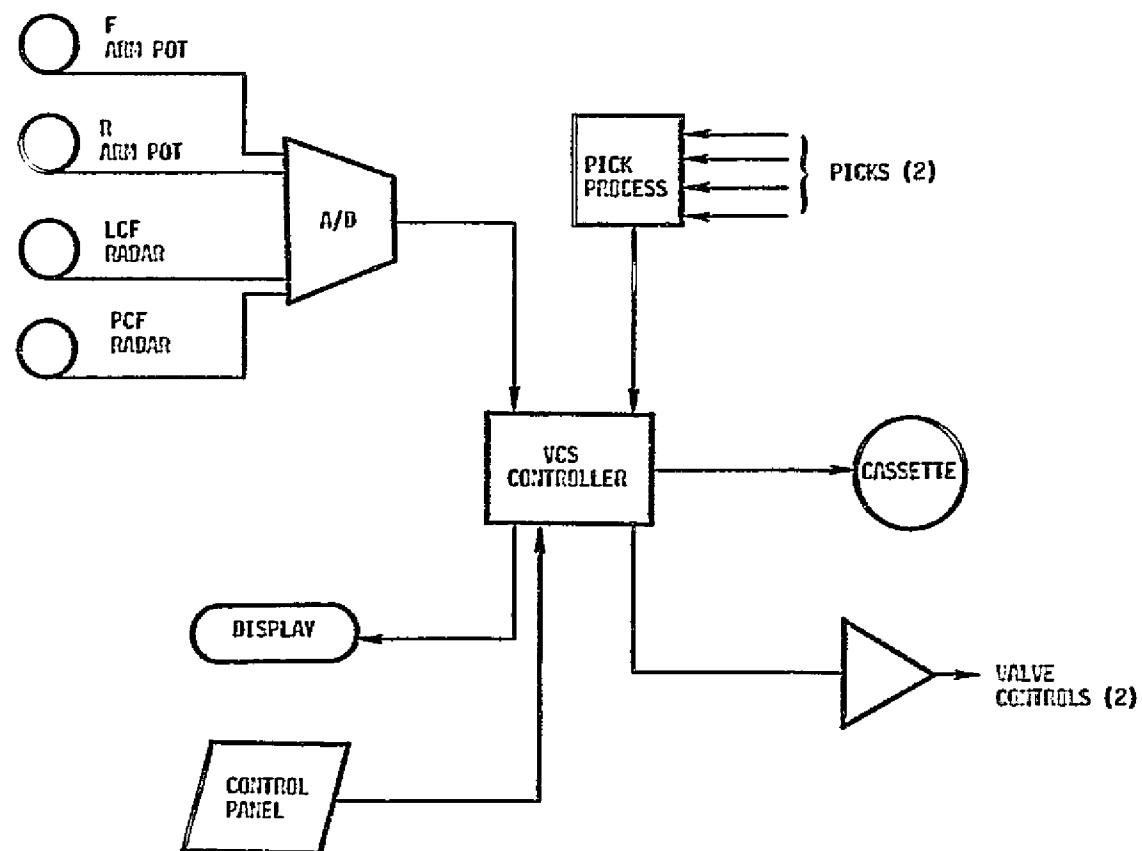








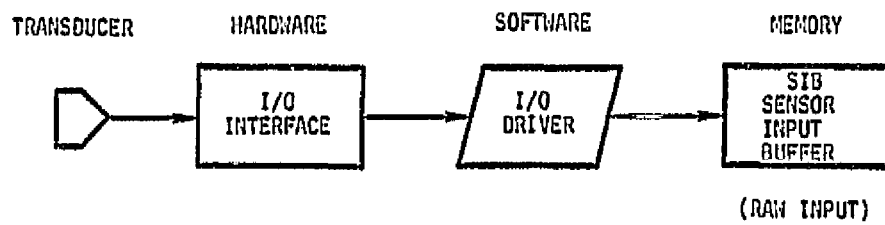




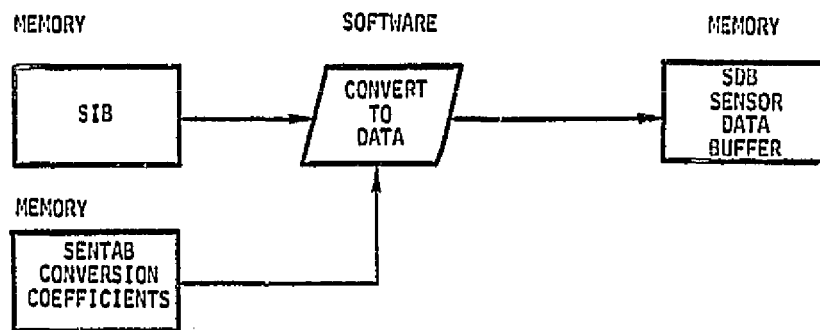
VCS [DEUCETON/KLF
OLD-BEN] HARDWARE

VCS DATA INPUT/CONVERSION

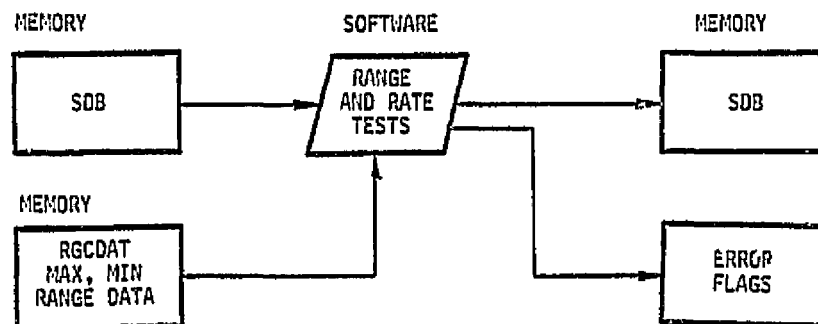
1) INPUT:

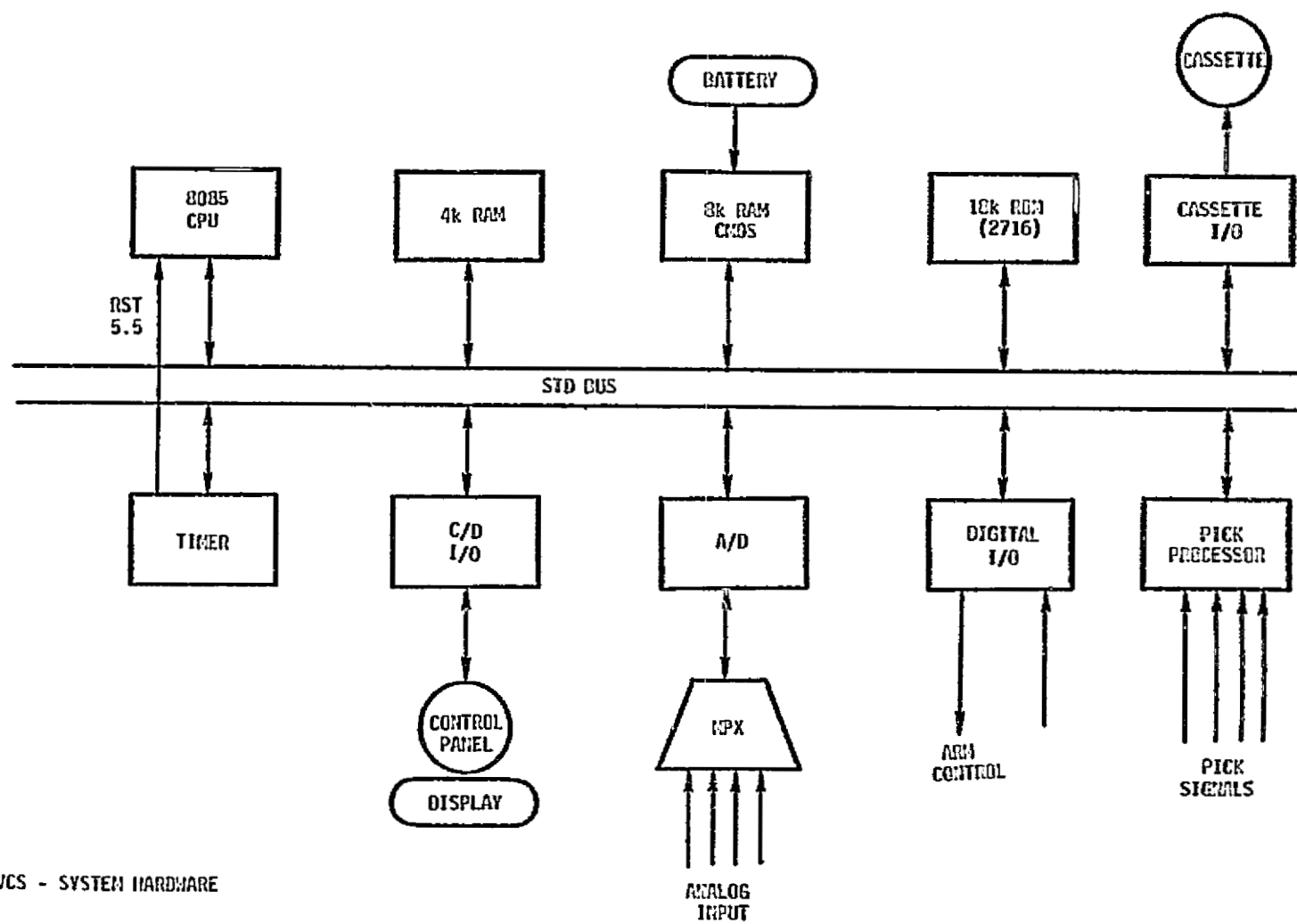


2) CONVERSION:



3) RANGE CHECK:





VCS - SYSTEM HARDWARE

VCS MEMORY ALLOCATION

	LOC	
MONITOR CODE	0000H 9FFH	CPU ROM
DATA TABLES (DEFAULT) MESSAGE CODES	A00H 1FFFFH	
VARIABLE SPACE COMMUNICATION BUFFERS	2000H	CPU RAM
STACK SPACE	2FFFFH	
NOT AVAILABLE	3000H 3FFFFH	-
VCS PROGRAM <ul style="list-style-type: none"> • EXECUTIVE • SUPERVISORS • EXECUTION MODULES • UTILITY FUNCTIONS • I/O DRIVERS • FORTRAN RTL • MISCELLANEOUS 	4000H 67FFH	ROM
SENSOR DATA SPECIAL VARIABLES	7000H 9000H	CMOS RAM

VCS MEMORY ALLOCATION

LOW ROM

MONITOR	0000H 9FF	} 8k
DEFAULT AND TABLES	0A00 0C00	
UNUSED AVAILABLE FOR DIAGNOSTICS, ETC.	0C01 1FFF	

LOW RAM

MONITOR SPACE	2000 20FF	} 4k
VARIABLES ~3 Kb	2100 27FF	
CONTAB, SENBUF	2800	
STACK	2FFF	

I/O PORT NUMBERS

Tape Recorder

08	OUT	RECHI	; Recorder Data HI byte
09	OUT	RECLC	; Recorder Data LO byte
0A	OUT	RECCTL	; Recorder Control
0B	IN	RECSTA	; Recorder Status

Picks	10	OUT	PICC1	; Pick Control 1
	11	IN	PICS1	; Pick Status 1
	12	OUT	PICC2	
	13	IN	PICS2	
	14	OUT	PICC3	These may not be required.
	15	IN	PICS3	
	16	OUT	PICC4	
	17	IN	PICS4	

Clock	20	Unit 0
	21	Unit 1
	22	Unit 2
	23	Control Port
	24	Counter Output Bit Port (3 bits)
	25	Gate Port

NBS	30
	31
	32
	33
	34

A/D Converter

80	OUT	MROUT	; Multiplexer out reg.
81	IN	MSRIN	; Multiplexer Status Port
82	OUT	MROUTC	; Multiplexer Out Control Port
83	OUT	MSRINC	; Multiplexer Control Port
84	IN	DATA1	; Data Port, MSB
85	IN	DATA 2	; Data Port, LSB

Keyboard Display Card

D0	IN	KDSW	; Keyboard/Display Switch Port
D0	OUT	KDDATA	; Keyboard/Display Data Port
D1	OUT	KDCNTL	; Keyboard/Display Control Port
DC	IN/OUT	CDATA	; EPROM/UART Console Data Port
DD	IN/OUT	CSTAT	; Console Status Port
DE	OUT	SCNTL	; System Control Port
03	OUT	ARMS	Arm Control
04	IN	SWTCH 1	; Front Panel Switches
05	IN	SWTCH 2	; Front Panel Switches
06	OUT	STAT1	; Front Panel Indicators
07	OUT	STAT2	; Front Panel Indicators

LINKAGE OF ASSEMBLY ROUTINES

ORG	Absolute origin
ASEG	Absolute segment
CSEG	Relocatable code segment
DSEG	Relocatable data segment
PUBLIC	Define symbols as public for use by other programs
EXTRN	Symbols from other programs

LINKAGE

Call name (A, B)

REG BC = Address (A)

DE = Address (B)

See INTEL FORTRAN User's Guide

VCS COMMON BLOCKS

/CLOCK/		CLOCK FLAG + COUNTERS
/COMBUF/		COMMUNICATION BUFFERS
/CONTAB/		CONVERSION TABLES
/CNTVEC/		CONTROL VECTORS
/CPANEL/		CONTROL PANEL ARRAY
/DEBUG/		DEBUG MASKS
/DEFAULT/	ROM	DEFAULT DATA
/DEFOPS/		
/DEFTAB/	ROM	DEFAULT DATA
/DEFMOS/		
/ERRFLG/		ERROR FLAG ARRAY
/ERRBUF/		
/IOFVEC/		I/O MASK VECTOR
/MESTAB/	ROM	MESSAGE TABLES I
/MESTB2/	ROM	MESSAGE TABLES II
/MESTB3/	ROM	MESSAGE TABLES III
/OPPARM/		OPERATION PARAMETERS
/OPSPNT/		OPERATION POINTERS
/OPSVEC/		OPERATION VECTOR
/PARAMS/		PARAMETER STORAGE
/PIKBUF/		PICK DATA STORAGE
/SENBUFF/		SENSOR INPUT, DATA BUFFER
/SENDEL/		SENSOR DELAY LINES
/SENMEM/	CMOS	NONVOLATILE STORAGE
/SENPNP/		
/SAVER/		
/WORK/		G.P. WORK SPACE
/OPPARM/		
NMAX (1)	=	NMXSEN
NMAX (2)	=	NMXERR
NMAX (3)	=	NMXSIR
NMAX (4)	=	NMXIPM
NMAX (5)	=	NMXOUT
NMAX (6)	=	NMXDSP
NMAX (7)	=	NMXIOF
NMAX (8)	=	NMXCNM,CTV
NMAX (9)	=	NMXDBG
NMAX (10)	=	NMXCBF

/OPPARM/

NMAX (11)	=	NMXTAP
NMAX (12)	=	NMXTIB
NMAX (13)	=	NMTPRM
NMAX (14)	=	NMTPPS
NMAX (15)	=	NMTPNT

/IOFVEC/

IOFVEC (1)	=	IOFSEN
IOFVEC (2)	=	IOFDED
IOFVEC (3)	=	IOFPCF
IOFVEC (4)	=	IOFLCF
IOFVEC (5)	=	IOFLAP
IOFVEC (6)	=	IOFRAP
IOFVEC (7)	=	IOFCID
IOFVEC (8)	=	IOFPKF
IOFVEC (9)	=	IOFPKR
IOFVEC (10)	=	IOFVCF
IOFVEC (11)	=	IOFVCR
IOFVEC (12)	=	IOFVOT
IOFVEC (13)	=	IOFDSP
IOFVEC (14)	=	IOFKEY
IOFVEC (15)	=	IOFCAS
IOFVEC (16)	=	IOFCRT
IOFVEC (17)	=	IOFCDU

/CNTVEC/

CNTVEC (1)	=	CTVCID	CID
CNTVEC (2)	=	CTVLCF	LAST CUT
CNTVEC (3)	=	CTVPKF	PICK F
CNTVEC (4)	=	CTVPKR	PICK R
CNTVEC (5)	=	CTVPCF	PAST CUT
CNTVEC (6)	=	CTVVCF	VALVE F
CNTVEC (7)	=	CTVVCR	VALVE R
CNTVEC (8)	=	CTVFRT	FRONT CNTL
CNTVEC (9)	=	CTVRER	REAR CNTL
CNTVEC (10)	=	CTVOUT	CNTL OUTPUT

/PARAMS/

DMAX	-	DRUM MAY
DMIN	-	DRUM MIN
ERRLMT	-	MAX# ERRORS FOR TRIP
SEAM	-	SEAM HEIGHT
PIKAVE	-	PICK AVERAGE #
PKSTEP	-	PICK STEP VALUE
DEDBDF	-	FRONT DEAD BAND
DEDBDR	-	REAR DEAD BAND
ARMLNG	-	ARM LENGTH POT GAIN = 1

/PARAMS/

DRMRAD - DRUM RADIUS
 LCPOS - LCF MOUNTING POS (DFD UNITS)
 PCPOS - PCF MOUNTING POS (DFD UNITS)
 CIDPOS - CID MOUNTING POS (DFD UNITS)
 STRLOC - FRONT TO REAR DIST. (DFD UNITS)
 DDFDCT - DFD GAIN FACTOR = 1
 SNDSTP - CMOS STEP UPDATE DISTANCE
 DBIASF - BIAS FRONT
 DBIASR - BIAS REAR
 SRKVAL - SENSOR RATE LIMIT VALUE
 TRGFCT - TRIG COMPENSATION FACTOR

/OPSVEC/

OPMODE - OPERATION MODE
 MODECT - CONTROL MODE
 RNGMSK - RANGE CHECK FLAG
 OPSENR - SENSOR DELAY FLAG
 RATECK - RATE CHECK FLAG
 STAT - SYSTEM STATUS
 FDVGN - FRONT CONTROL GAIN
 FDCAL - FRONT TRIG FACTOR
 RDVGN - REAR CONTROL GAIN
 RDCAL - REAR TRIG FACTOR

PROGRAM OPERATION PARAMETERS

/OPPARM/ ICMPTNT(10), NMXVEC(16), NMXIPX(6)

ICMPTNT(1)	7	CID MAP CONTRU SET INTO MESSAGE VECTOR
(2)	4	LCF
(3)	8	PKF
(4)	9	PKR
(5)	3	PCF
(6)	10	VCF
(7)	11	VCR
(8)	18	FRT
(9)	19	RER
(10)	20	OUT

NMXVEC(1)	8	NMXSEN
(2)	8	NMXERR
(3)	0	NMXSIR
(4)	6	NMXIPM
(5)	0	NMXOUT
(6)	8	NMXDSP

NMXVEC (7)	20	NMXIOF	
(8)	10	NMXCNM, CTV	
(9)	9	NMXDBG	
(10)	8	NMXCBF	
(11)	128	NMXTAP	
(12)	8	NMXKTB	
(13)	20	NMXPRM	
(14)	0	NMXOPS	
(15)	8	NMXPNT	
(16)	0	DUMMY	
NMXIPX (1)	17	I/O	NUMBER OF PROMPTS TO ASK
(2)	10	CONTROL	FOR EACH TYPE
(3)	4	OPERATION	
(4)	10	PARAMETERS	
(5)	1	CMOS MEM TABLE	
(6)	32	SENSOR TABLES	

ASCII TABLES

/MESTAB/
ERRMES: CHARACTER*4 (20)

1	\$SEN
	\$DPD
	\$PCF
	\$LCF
5	\$LAP
	\$RAP
	\$CID
	\$PKF
	\$PKR
10	\$VCF
	\$VCR
	\$VOT
	\$DSP
	\$KEY
15	\$CAS
	\$CRT
	\$CDU
	\$FRT
	\$RER
20	\$COT

DBGMES: CHARACTER*4 (20)

1	@DBG
	OUT?
	INP?
	DAT?
5	ITZ?
	TAP?
	I/O?
	CLK?
	MEM?
10	DUMP
	END?
	@SYS
	PORT
	CHNL
15	UP
	DOWN
	MAN
	MODE
	WRIT
20	ADR?

RUNMES: CHARACTER*4 (20)

1	?I/O
	?CNT
	?OPS
	?VRB
5,6	*RUN-OK*
7,8	*DEBUG *
9,10	?CALBRAT
	?PAUSE ?
	MASK
	PARM
15	ERR
	?-STOP-?
	-EXIT-
20	##

DSPCODE: CHARACTER*1 (10)

1:	\$ * # & ?
6:	@ ? = /

PARAMETER MESSAGE TABLE

PARMES (20)

CHARACTER*4

/MESTB2/

1	DMAX
2	DMIN
3	ERLM
4	SEAM
5	PAVE
6	PKSP
7	DBDF
8	DBDR
9	LENG
10	DRAD
11	LCPS
12	PCPS
13	CDPS
14	FLOC
15	DFCT
16	DSTP
17	FBAS
18	RBAS
19	SKRV
20	TRGF

/MESTB3/
OPSMES: CHARACTER*4 (10)

1: OPER
MODE
RANG
OSDL
5: RATE
STAT
FDVG
FCAL
RDVG
RCAL

OPERATION VARIABLE VALUES

OPMODE:	-1	REBOOT
	0	STOP
	1	RUN
	2	DEBUG
	3	HALT
	4	MANUAL
	5	CALIBRATE
MODECT:	0	NO CONTROL
	1	MODE II
	2,4	MODE I
	3	REAR SLAVE
RANG:	0	NO CHECK
	1	RANGE CHECK
OPSENR:	0	DIRECT
	1	DELAYED (PREVIEW)
RATECK:	0	NO CHECK
	1	RATE CHECK
STATUS:	0	SYSTEM COLD START
	1	RUNNING

SENSOR INDEX POINTER

SINPNT:

1	DFD
2	PCF
3	LCF
4	LAP
5	RAP
6	CID
7	PKF
8	PKR

USED FOR CONTROL OF A/D AND CONVERSION AND RANGE
CHECK TABLE ENTRIES

POINTS INTO SIB OR SDB ENTRIES FOR APPROPRIATE VALUE

VCS ERROR CODES

ERROR(1) = I (SINPNT) FOR SENSOR INPUT RANGE ERROR POSITIVE
I + 10 FOR NEGATIVE

ERROR(2) = 1 IF 39 VOLT FAILURE

ERROR(3) = 1 CAL SWITCH $\overline{\text{RUN}}$ FROM RUN MODE

ERROR(4) RATE CHECKING
= 1 LCF ERROR
= 10 PCF
= 11 BOTH

ERROR(5) = N; CASSETTE ERROR (DUMMY(2))
ERROR FLAG BYTE = STATUS FROM CASSETTE

ERROR(6) CMOS DATA NOT READY MODE I MODECT = 2 or 4

PROGRAM MODULES

VCS SUPERVISOR PROGRAM

Main program: VCSSUP

Main program calls supervisor routines based on flags and of modes.

Inputs: Control information buffers

Outputs: Nothing

Calls: SUPINP, SUPOUT, SUPCNT, SUPDBG, SUPITZ, SUPCAL,
SUPERR, SUPEXT, SUPHLT

Destroys: Nothing

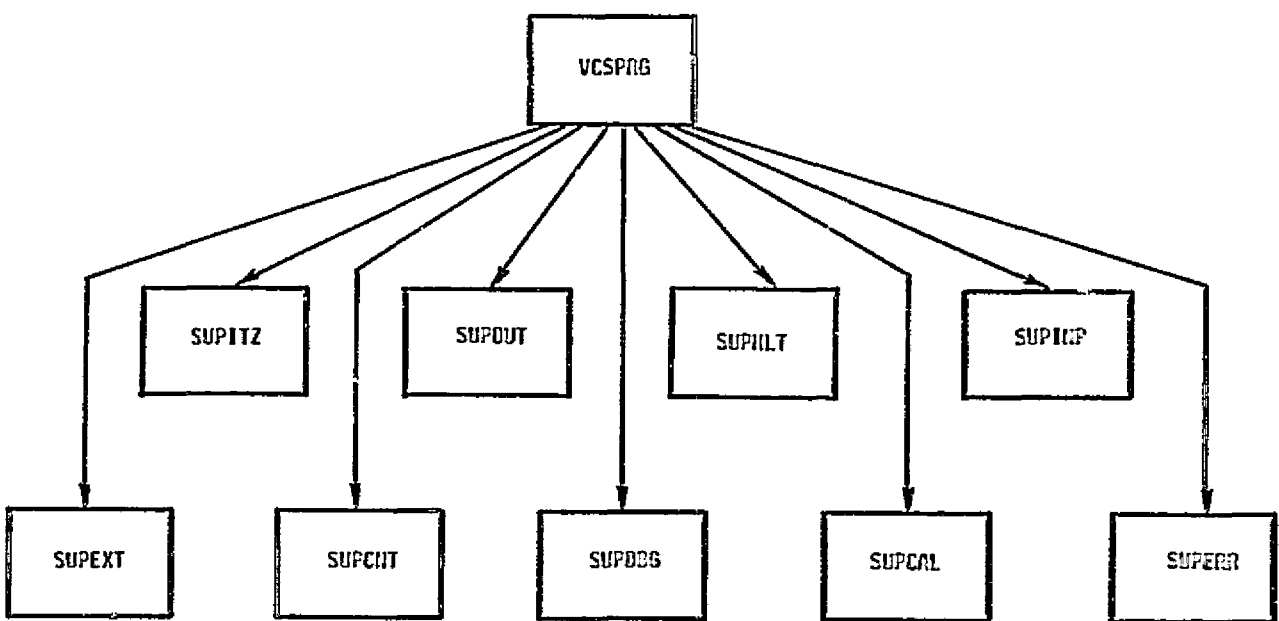
Modifies: Nothing

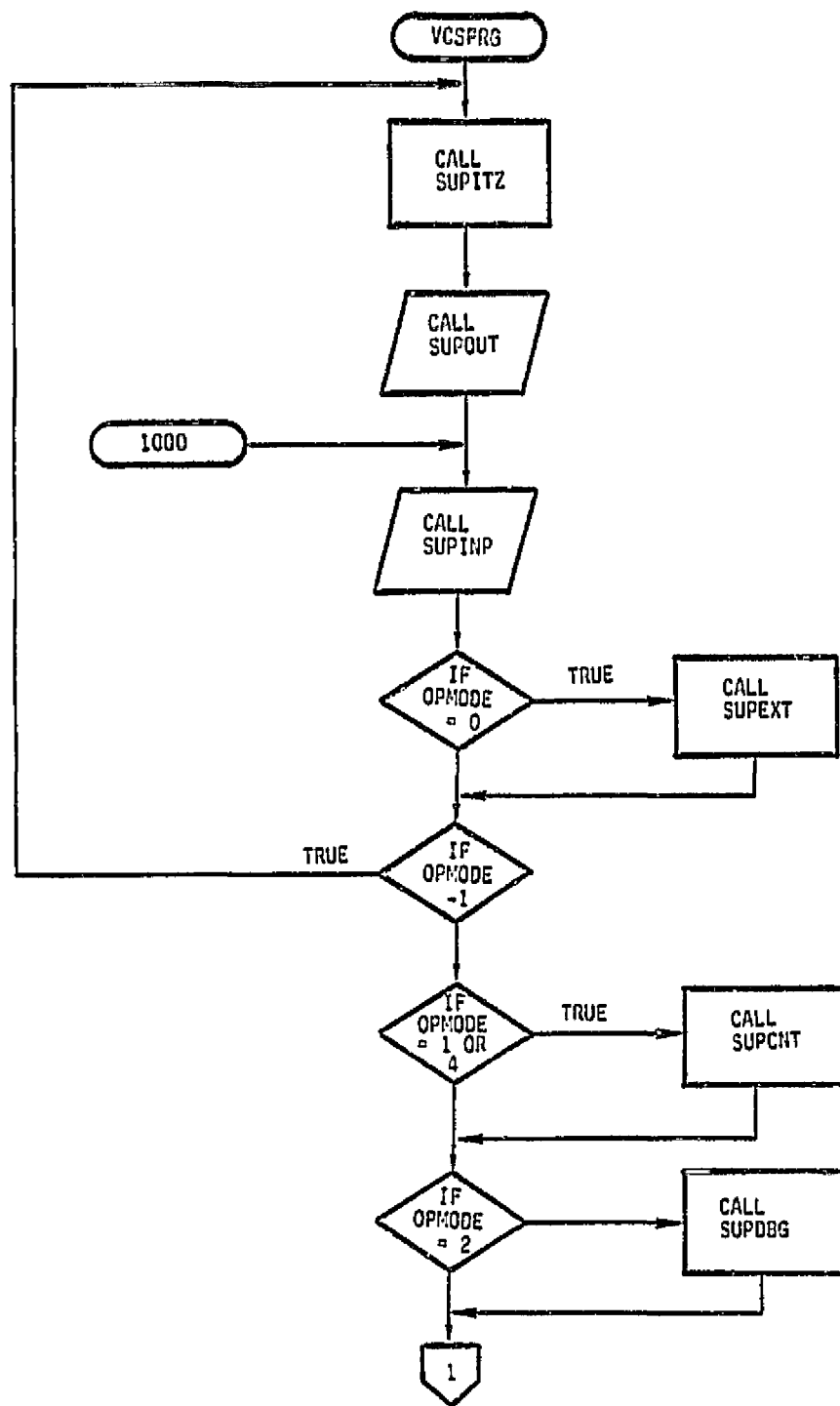
Language: FORTRAN 80

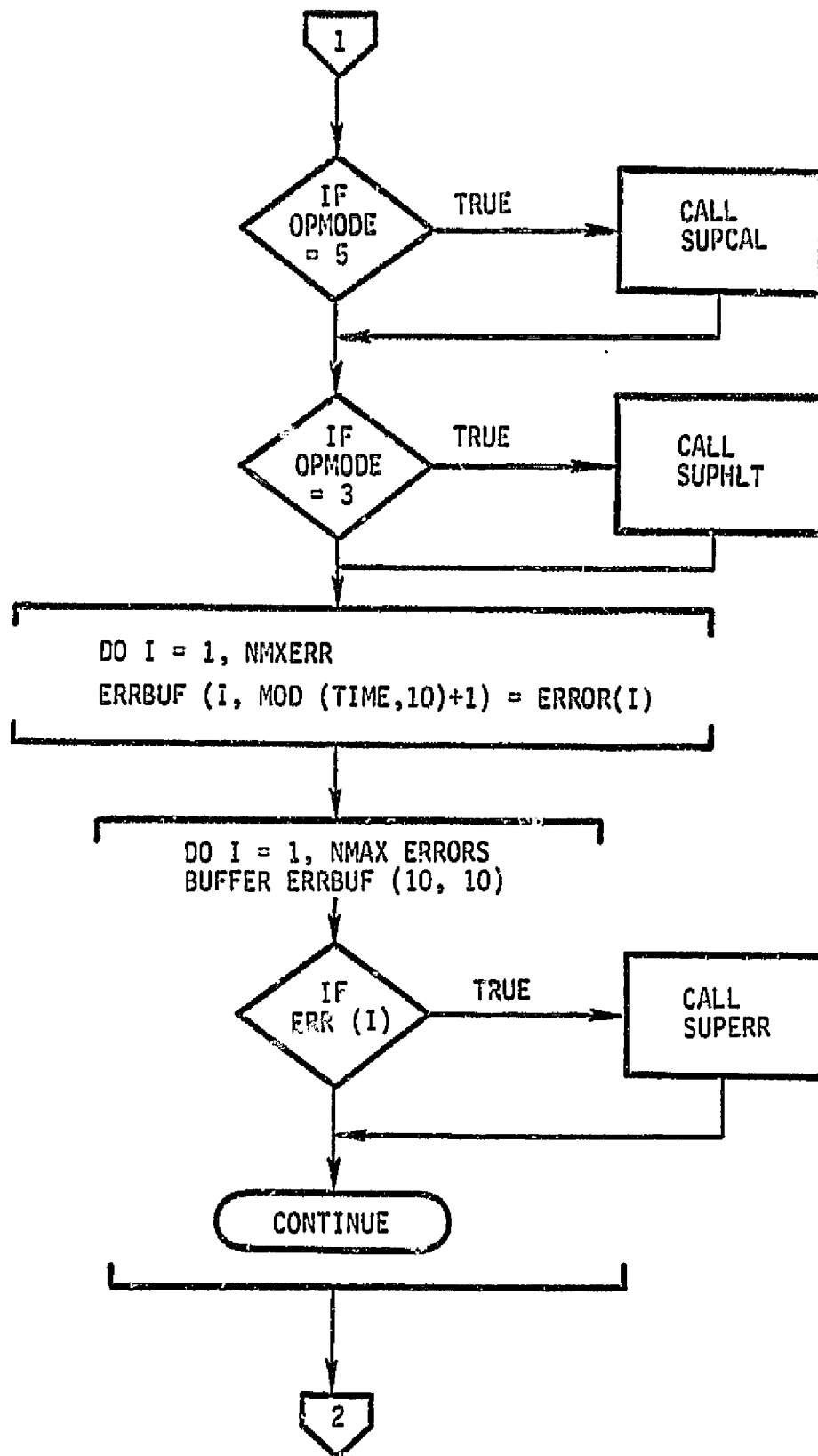
L-V-A: Version 1, Level 1
Roger B. Fish

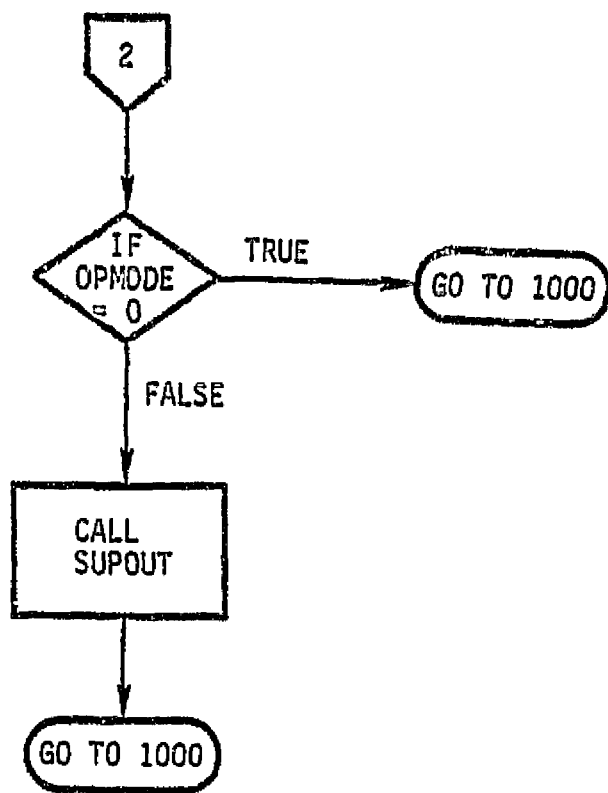
MODULE: VCSPRG

VCSPRG is the main program executive module of the VCS software. The main program supervises operation of the VCS by calling the appropriate supervisor routines to perform the required operations. VCSPRG continuously monitors operation parameters (OPMODE, ERROR(I)) and calls the relevant routines to handle the condition. (For example, DEBUG, RUN, INITIALIZE, ERROR PROCESSING.) Communication between modules in the VCS software, except for certain I/O drivers, is via global COMMON blocks. VCS operates by continuously scanning the mode and status data and looping through its test code. The routine first inputs all sensor and control variables and then tests for supervisor routines to be called. Unless the supervisor causes a program termination, the system outputs are then updated and the process repeats.









SUPERVISOR SUBROUTINES

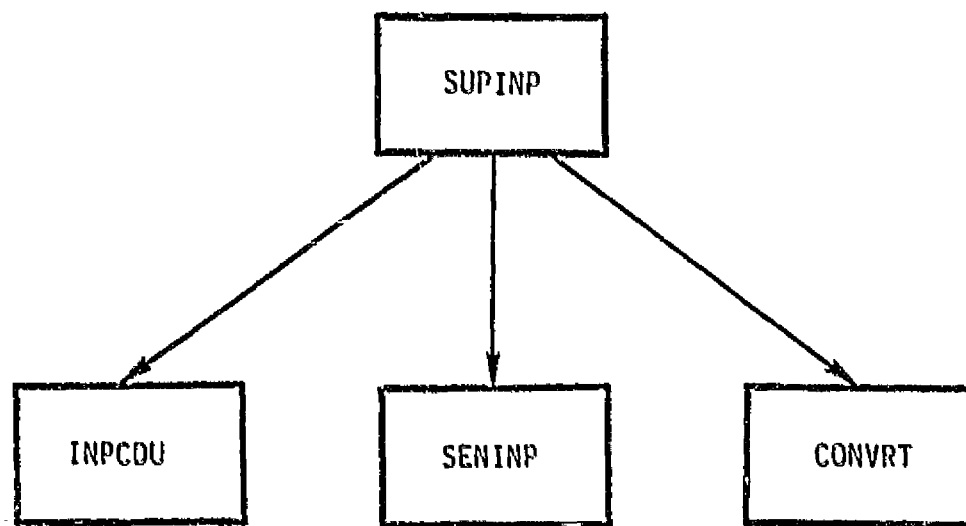
SUPINP	Input supervisor (sensors)
SUPOUT	Output supervisor
SUPCNT	Control supervisor
SUPDEG	Debug supervisor
SUPITZ	Initialization routine
SUPERR	Error handler
SUPEXT	Exit routine
SUPCAL	Calibration routines (restart)
SUPHLT	Program hard pause

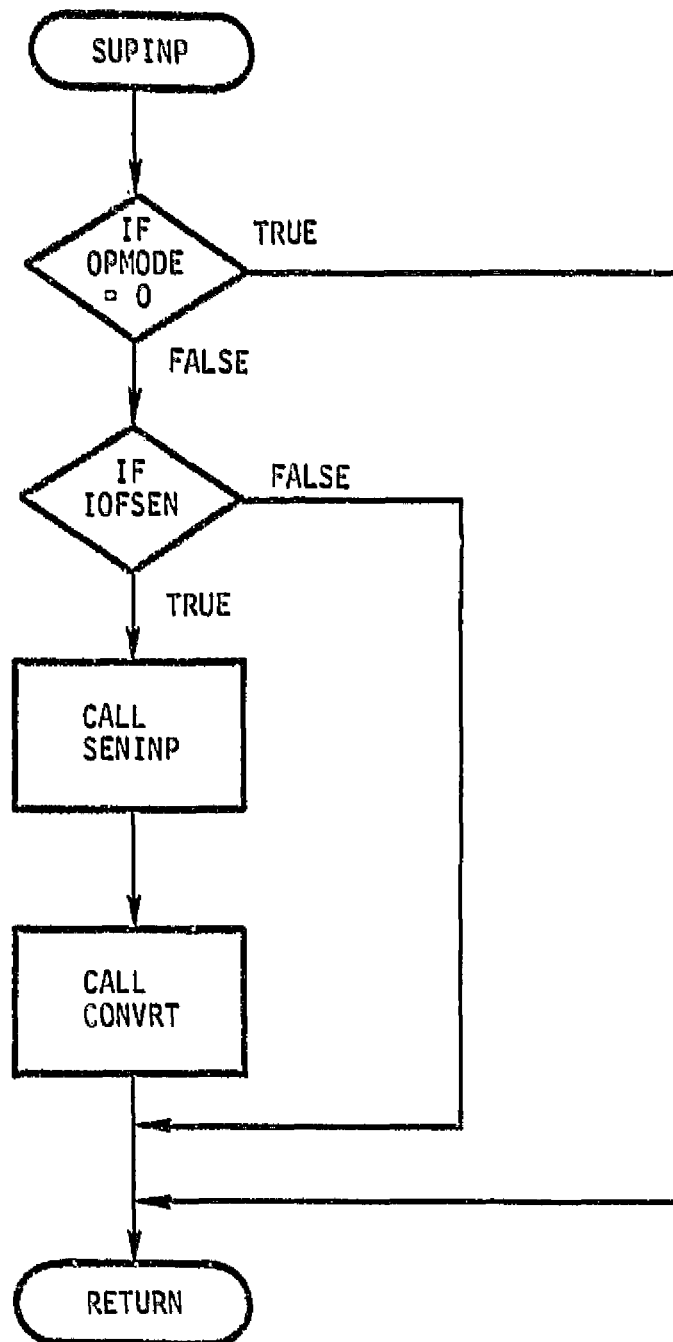
SUPERVISOR CONTROL ROUTINE

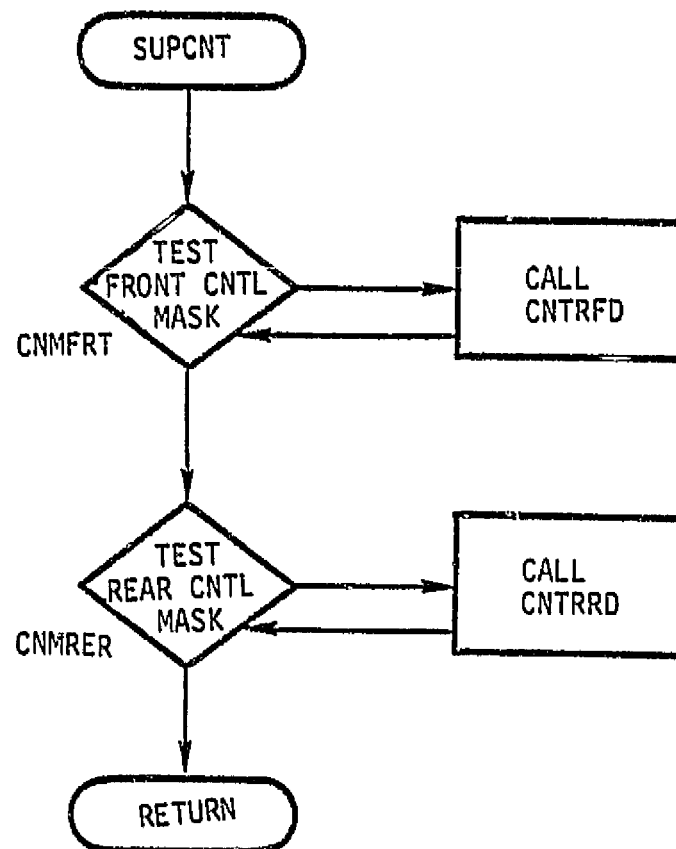
SUBROUTINE:	SUPINP
SUPINP	Control input to the VCS
Inputs:	I/O control buffer
Outputs:	Nothing
Calls:	SENINP, INPCDU, CONVRT
Destroys:	Nothing
Modifies:	Sensor input buffer
Language:	FORTRAN 80
L-V-A:	Version 1, Level 1 Roger B. Fish

MODULE: SUPINP

SUPINP is the supervisor routine which inputs sensor and control data to the VCS. The routine reads the operator control panel switches and then tests if sensor I/O is enabled by an I/O flag. If the flag, IOFSEN, is true, the sensor input routines SENINP for LCF, PCF, etc. is called. The sensor conversion routine CONVRT is called to convert the raw sensor data into appropriate engineering units.







SUPERVISOR CONTROL ROUTINE

Subroutine: SUPOUT

SUPOUT controls outputs from the VCS

Inputs: I/O control vector

Outputs: Nothing

Calls: CNTOUT, SAVDAT, FORMAT, CLOCK, INTREN,
 OUTDSP, OUTPUT

Destroys: Nothing

Modifies: Nothing

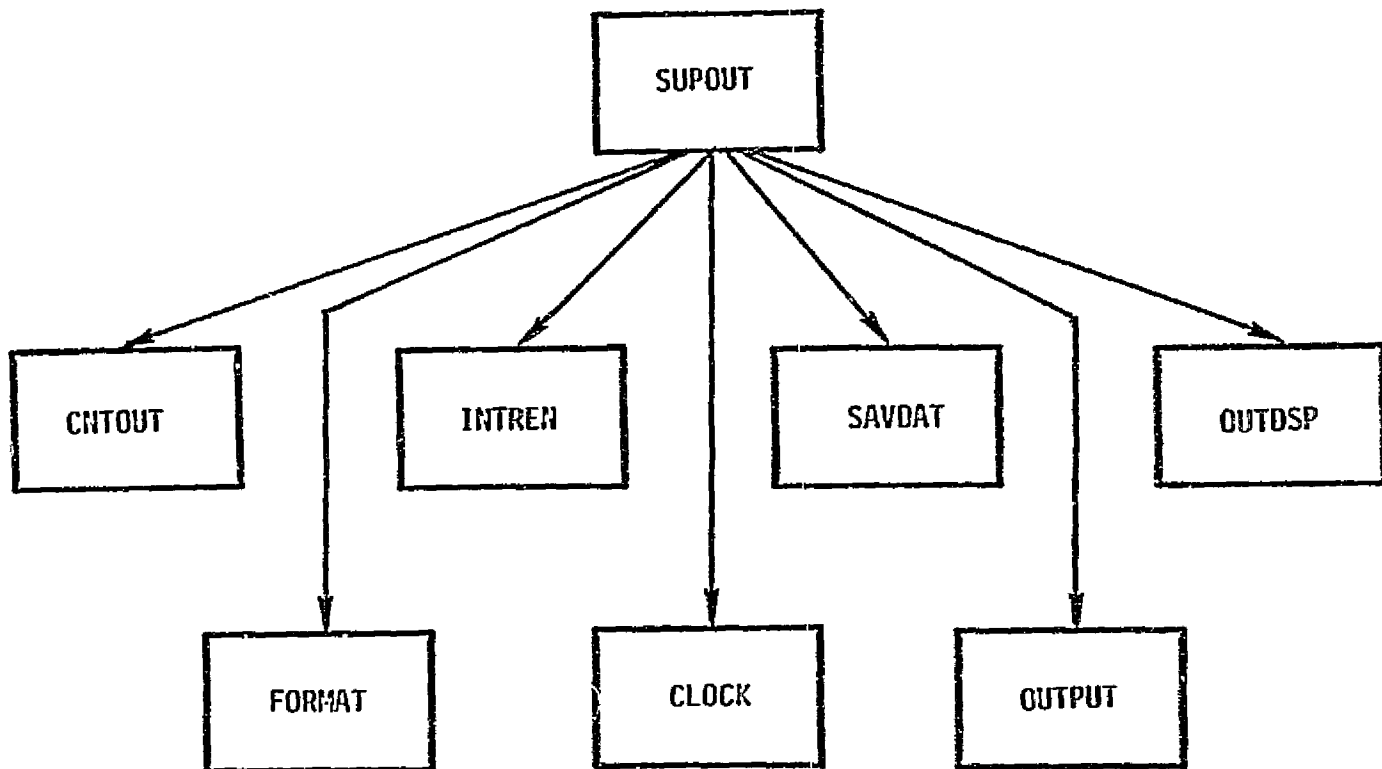
Language: FORTRAN 80

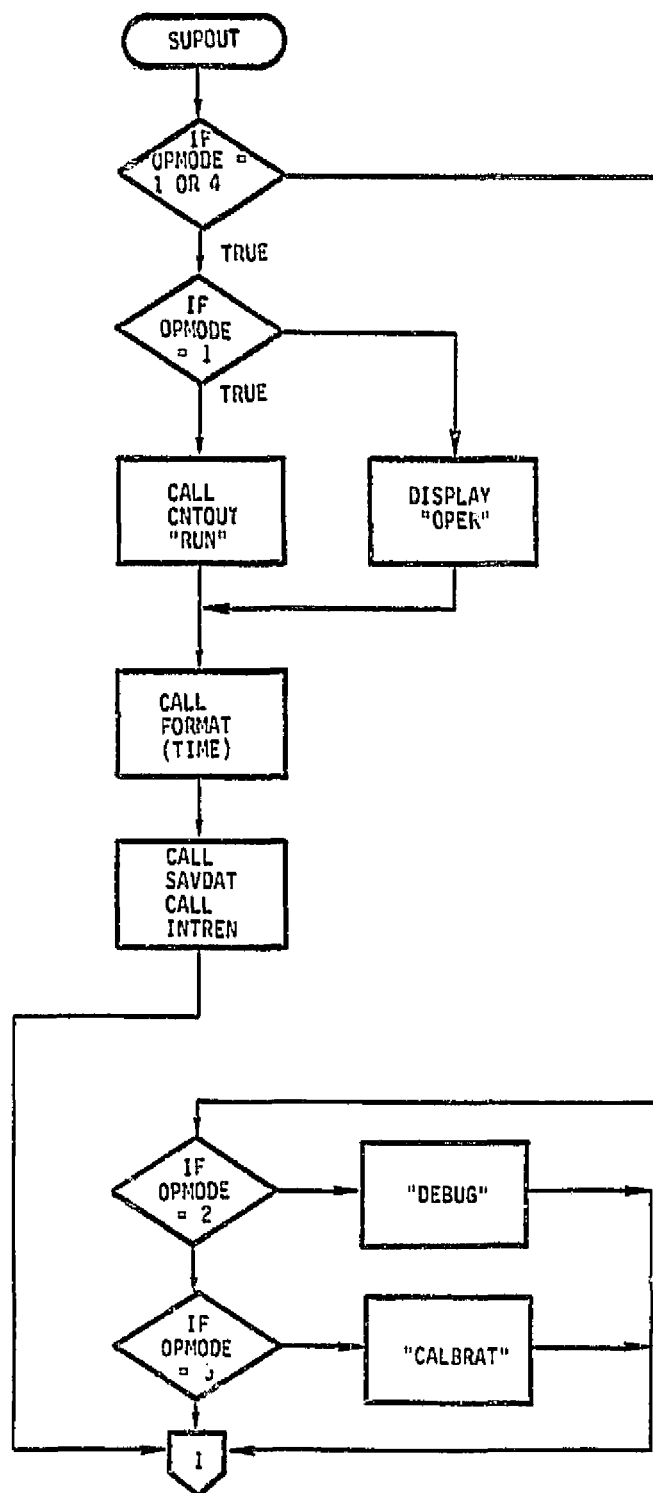
L-V-A: Version 1, Level 1
 Roger B. Fish

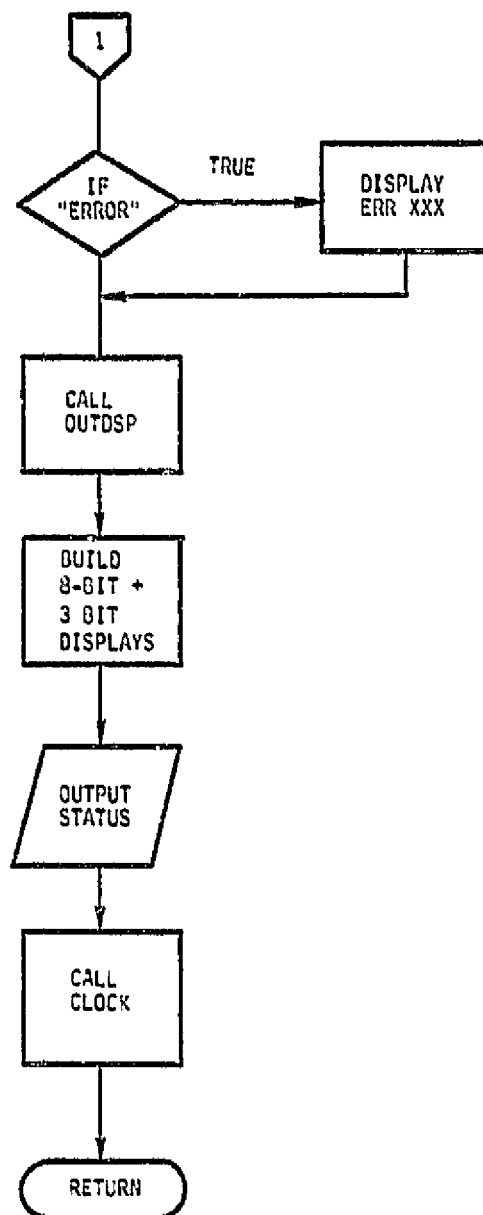
MODULE: SUPOUT

SUPOUT is the supervisor routine which outputs data from the VCS to the various control devices, control indicators, and operator control panel.

The operation mode value (OPMODE) is tested and an appropriate message is displayed on the display panel, and appropriate output routines are called. Display information for the control display is formatted and output. Finally, CLOCK is called for system synchronization.







SUPERVISOR CONTROL ROUTINE

Subroutine: SUPCNT

SUPCNT controls VCS control algorithm use and signal generation.

Inputs: Operation control vectors

Outputs: Nothing

Calls: CNTREFD, CNTRRD

Destroys: Nothing

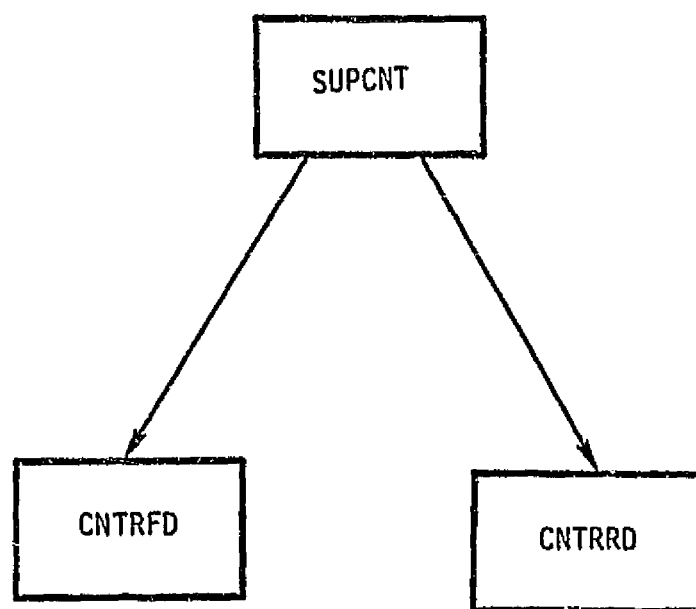
Modifies: Nothing

Language: FORTRAN 80

L-V-A Version 1, Level 1
Roger B. Fish

MODULE: SUPCNT

SUPCNT is the VCS control algorithm supervisor routine. This routine tests the control operation flags and control words and calls the appropriate routine to control the front, rear, or output the valve control signals.



SUPERVISOR CCNTROL ROUTINE

Subroutine: SUPDBG

SUPDBG is the debug operation routine

Inputs: Debug control vector

Outputs: None

Calls: DBUGCI, DBUGCO, DBUGIN, DBUGOT, DBUGTP,
ITZINP, DBUGCK, DBUGME, DBUGIO, INTRDS

Destroys: Nothing

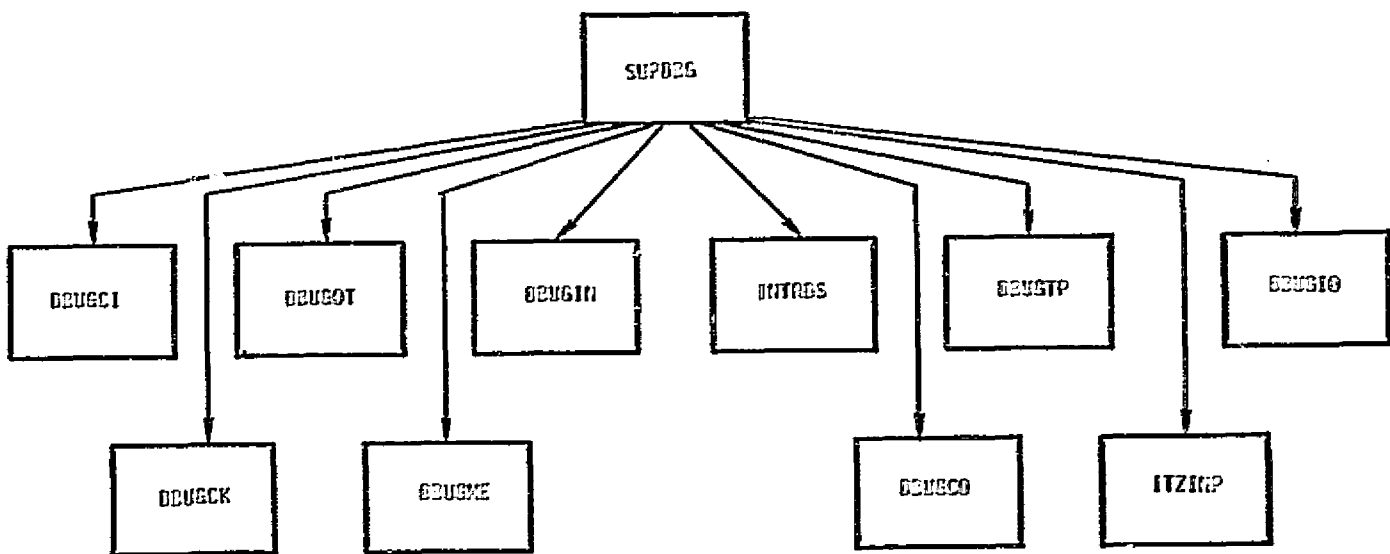
Modifies: Nothing

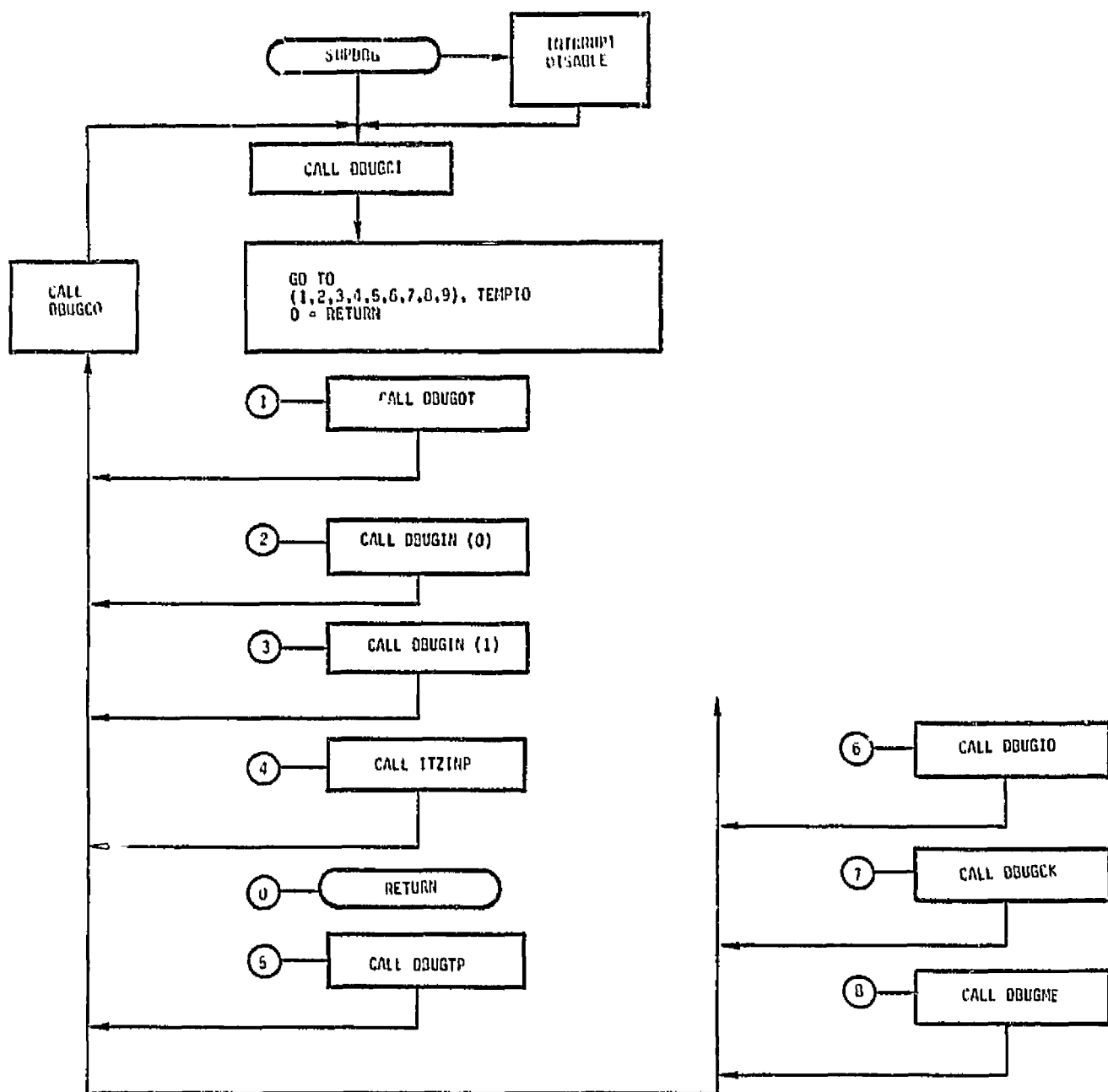
Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish

MODULE: SUPDBG

SUPDBG is the debug interaction supervisor. SUPDBG is called by *MAIN* when OPMODE is set to the debug value. SUPDBG then calls the debug control input routine DEBUGCI which prompts the user for the appropriate control input to select one of the debug options. When debug is complete, the routine returns to *MAIN*. SUPDBG can also call the interactive initialization routine ITZINP if the operator wishes to modify any variables.





SUPERVISOR CONTROL ROUTINE

Subroutine: SUPITZ

SUPITZ initializes the program variables buffers and control variables.

Inputs: None

Outputs: System initialized

Calls: INTRDS, INTRCN, OUTDSP, AIOITZ, ITZINP,
INPCDU, RESTRT, TAPITZ

Destroys: Nothing

Modifies: Input buffers, parameters, interrupts,
control information

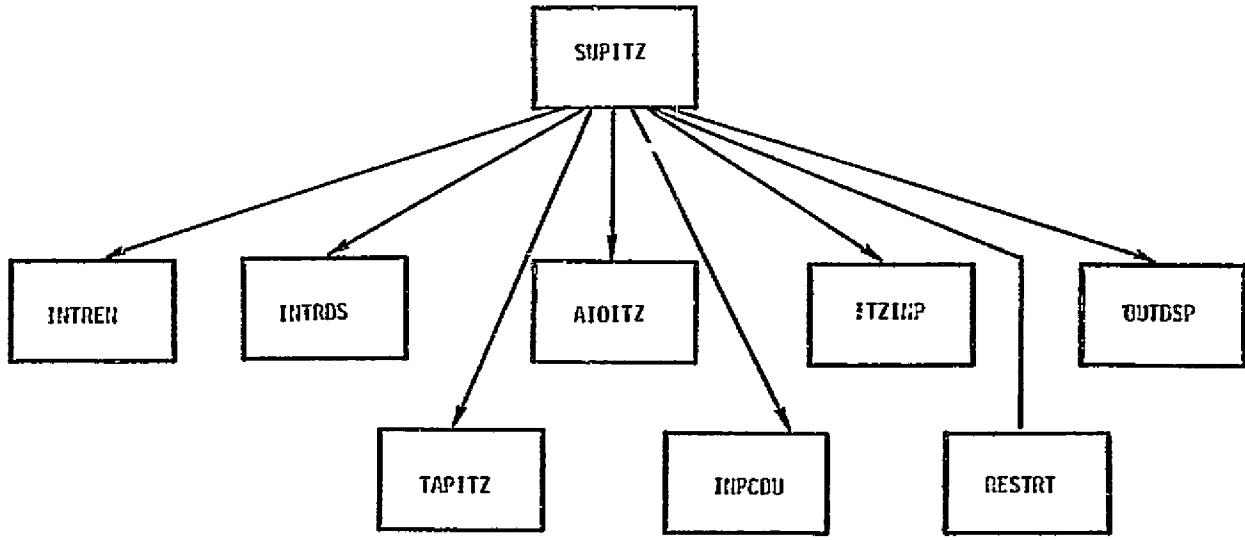
Language: FORTRAN 80

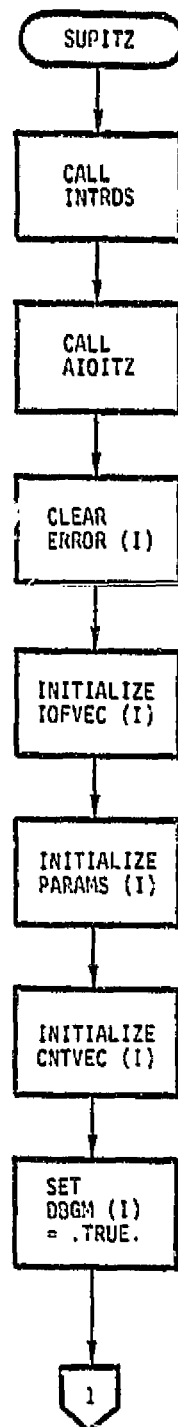
L-V-A: Version 1, Level 1
Roger B. Fish

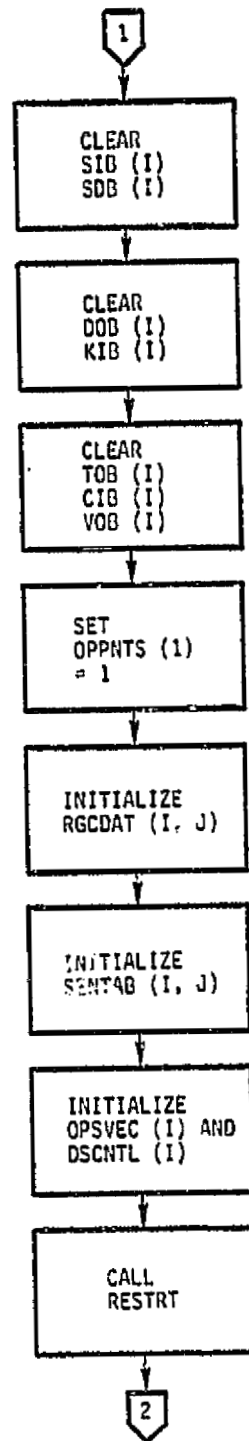
MODULE: SUPITZ

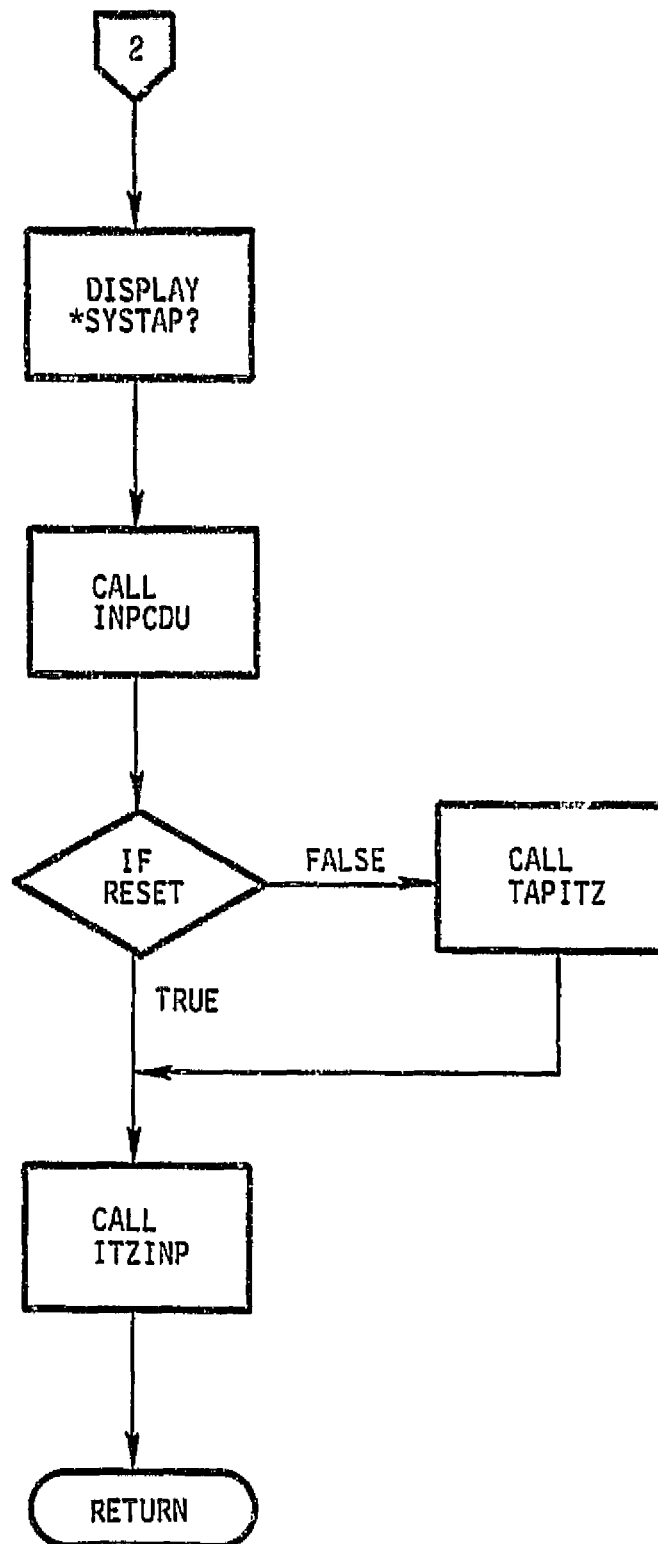
SUPITZ is the program initialization routine. SUPITZ takes care of initializing I/O interfaces via AIOITZ, clears I/O buffers to zero, and loads common blocks from the default ROM areas. The default values are copied into RAM memory so that the values can be modified via routine ITZINP.

SUPITZ also calls INTRDS to disable system interrupts.









SUPERVISOR CONTROL ROUTINE

Subroutine: SUPERR

 SUPERR processes error condition flags and outputs message.

Inputs: Error flag vector

Outputs: D/C error message

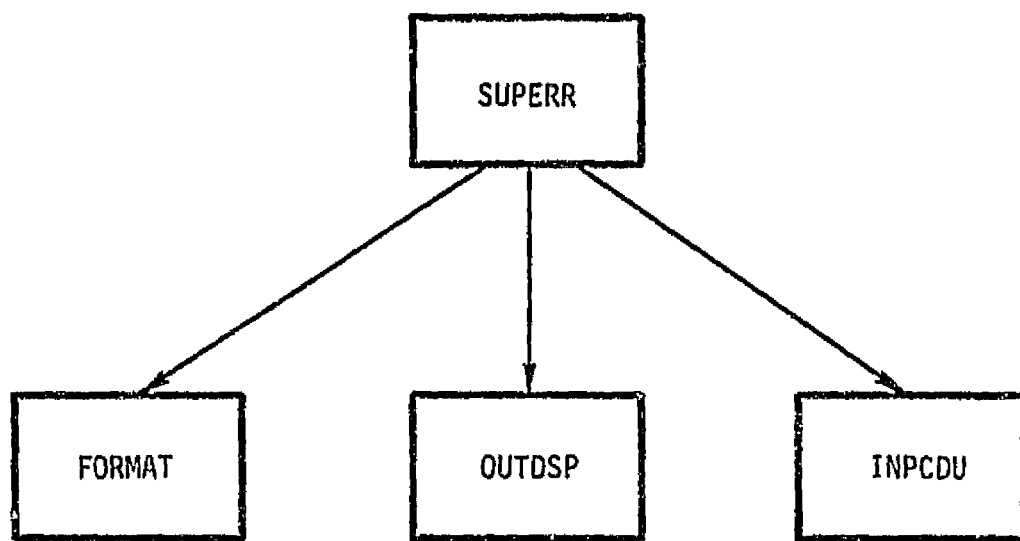
Calls: OUTDSP, INPCDU, FORMAT

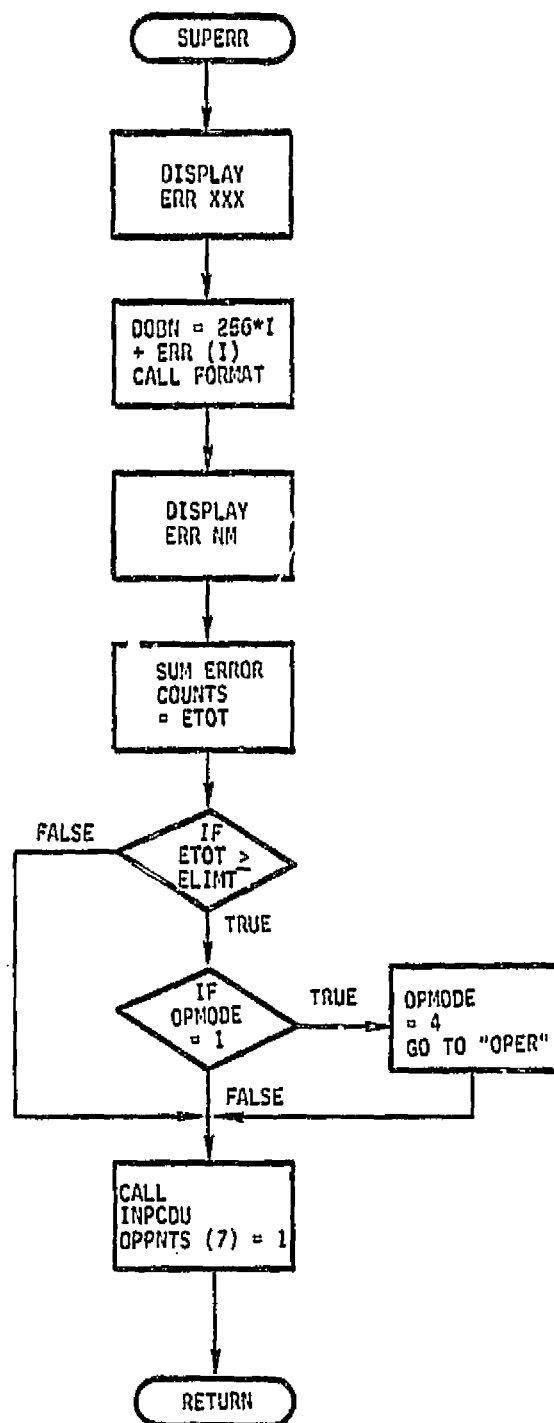
Destroys: Error flags

Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish





SUPERVISOR CONTROL ROUTINE

Subroutine: SUPEXT

SUPEXT exits from the VCS program after closing data base.

Inputs: None

Outputs: None

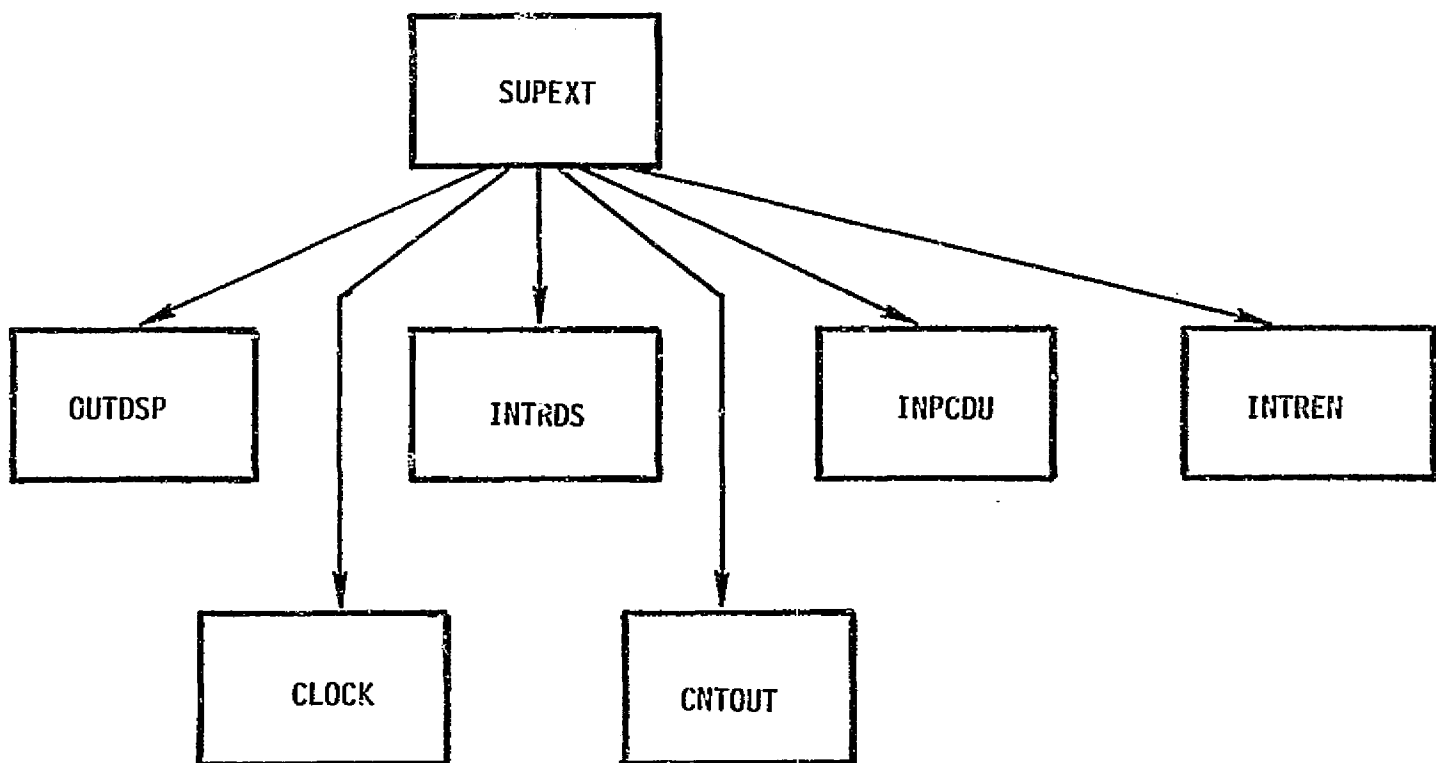
Calls: CLOCK, INTRDS, INPCDU, OUTDSP, CNTOUT,
 INTREN

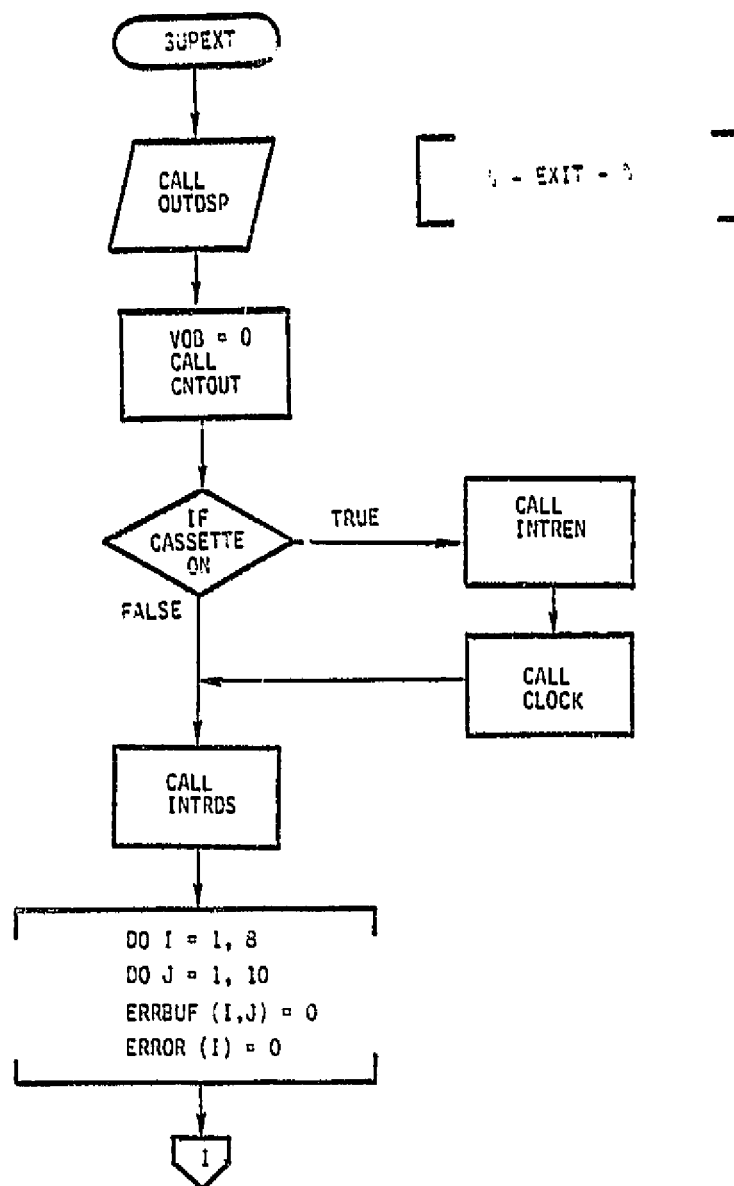
Destroys: S1B, SDB, ERRORS

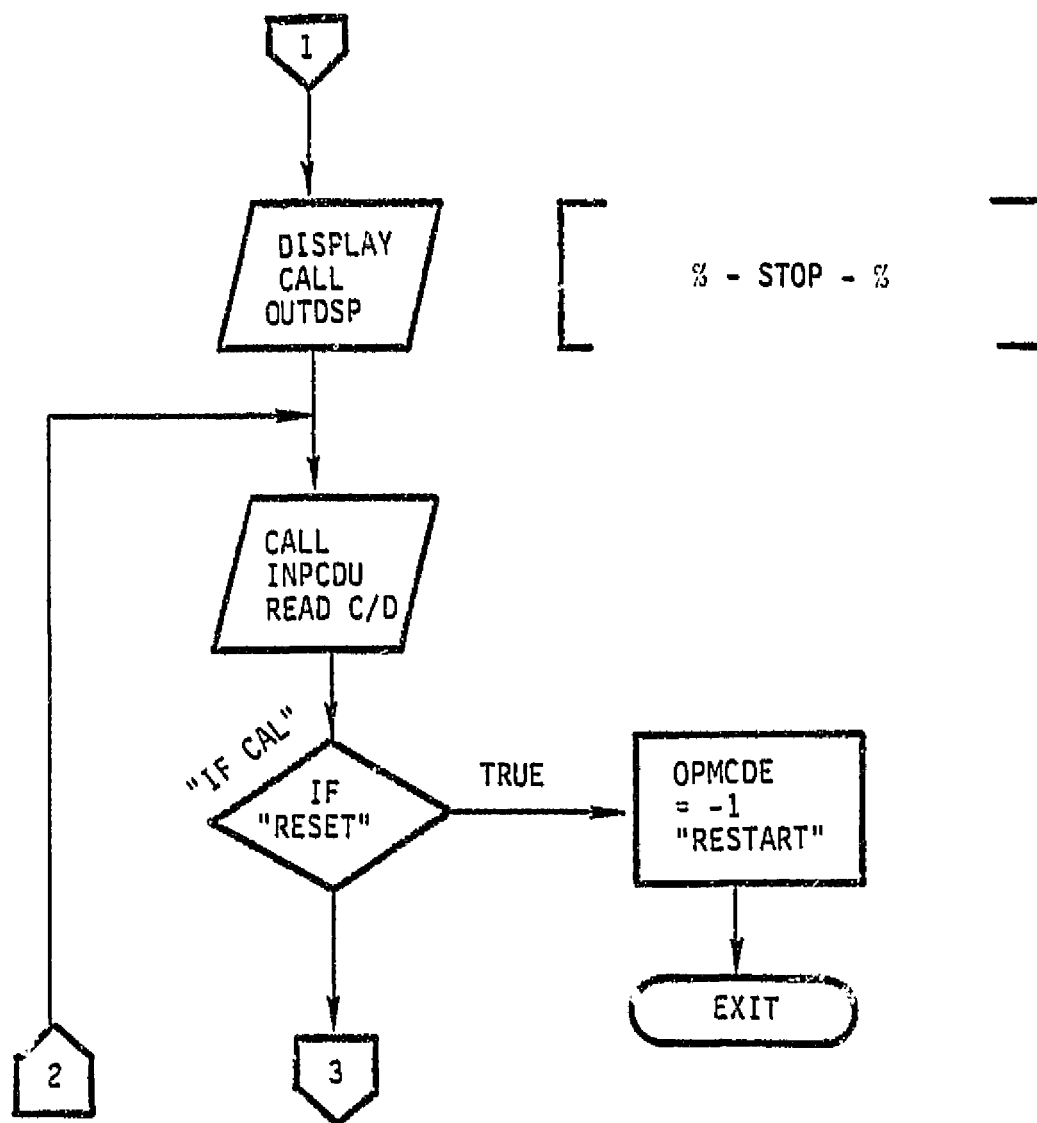
Modifies: Pointers, control words

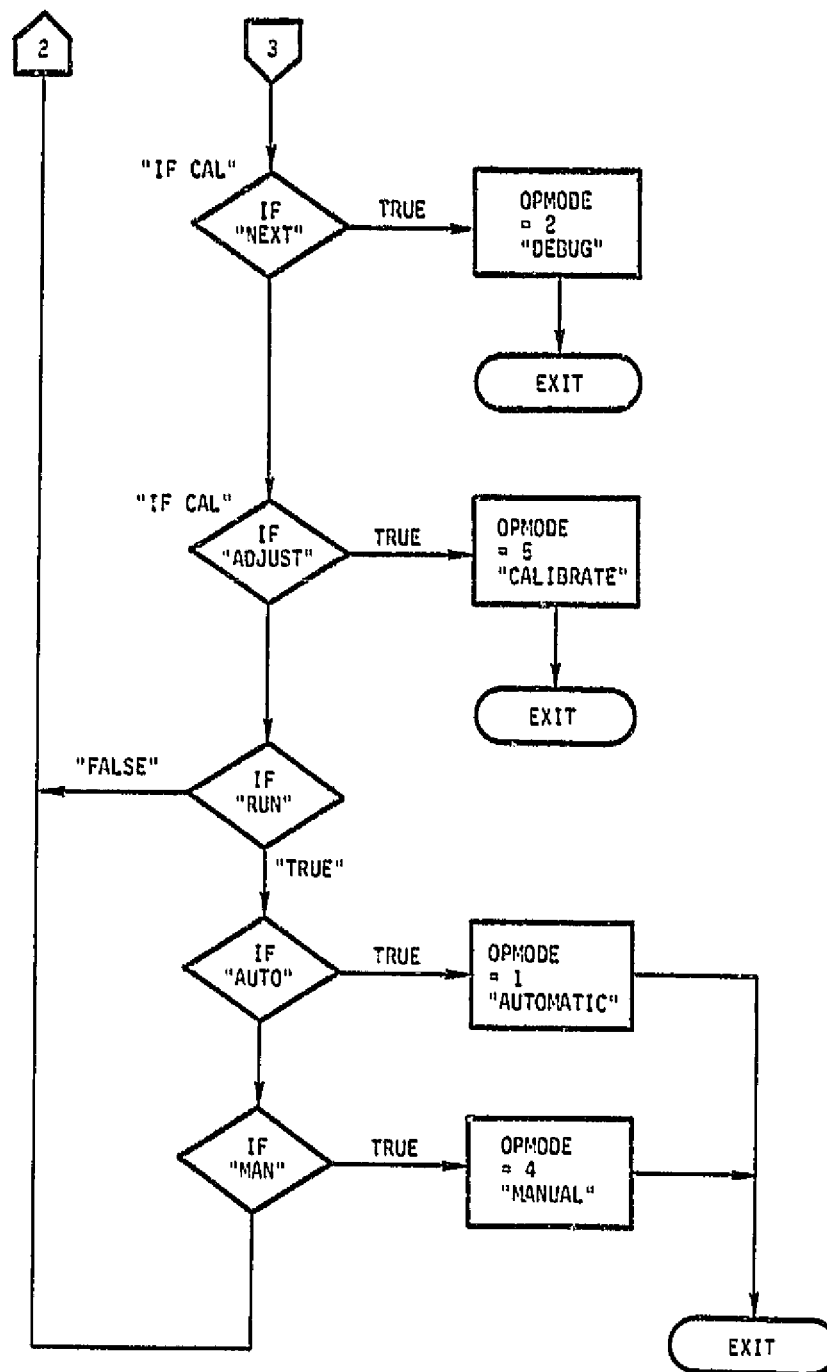
Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish









SUPERVISOR CONTROL ROUTINE

Subroutine: SUPCAL

SUPCAL calls the system restart (warm start) routine.

Inputs: None

Outputs: OPMODE

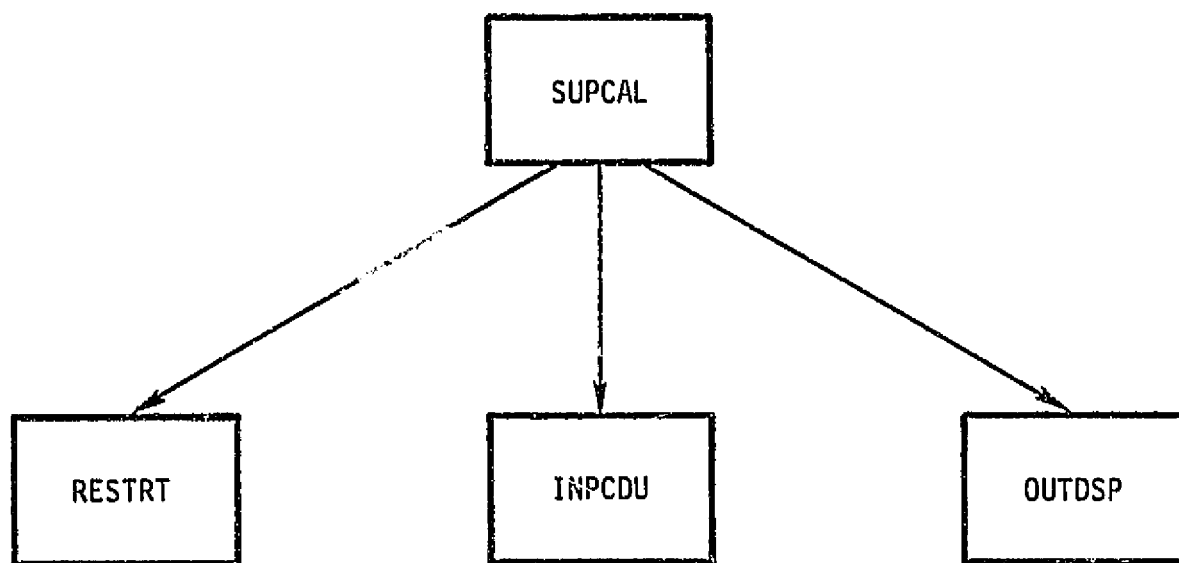
Calls: RESTRT, INPCDU, OUTDSP

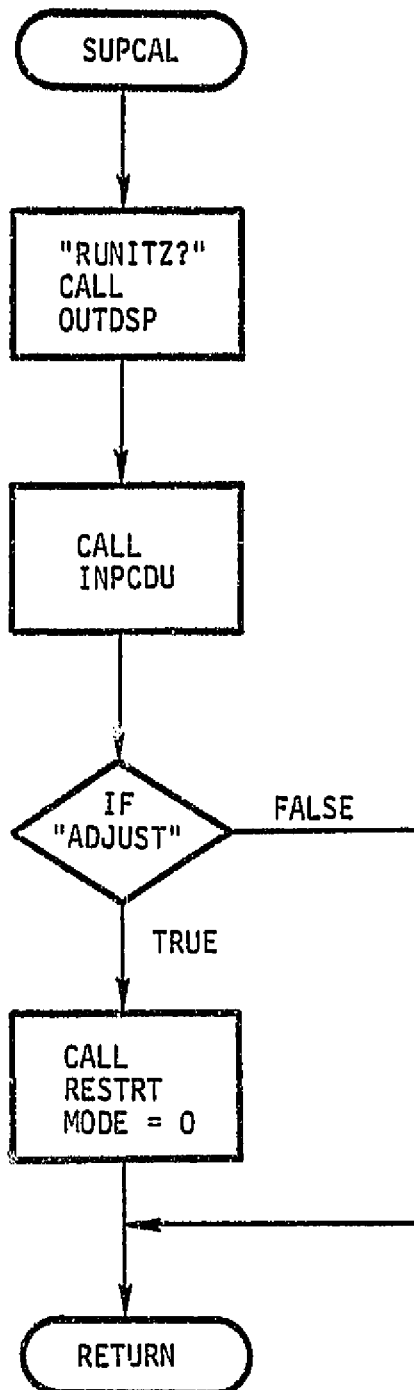
Destroys: Nothing

Modifies: OPSVEC, DOB

Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish





SUPERVISOR CONTROL MODULE

SUBROUTINE: SUPHLT

SUPHLT stops VCS program operation by stopping clock interrupts and blocks program flow.

Inputs: None

Outputs: OPMODE

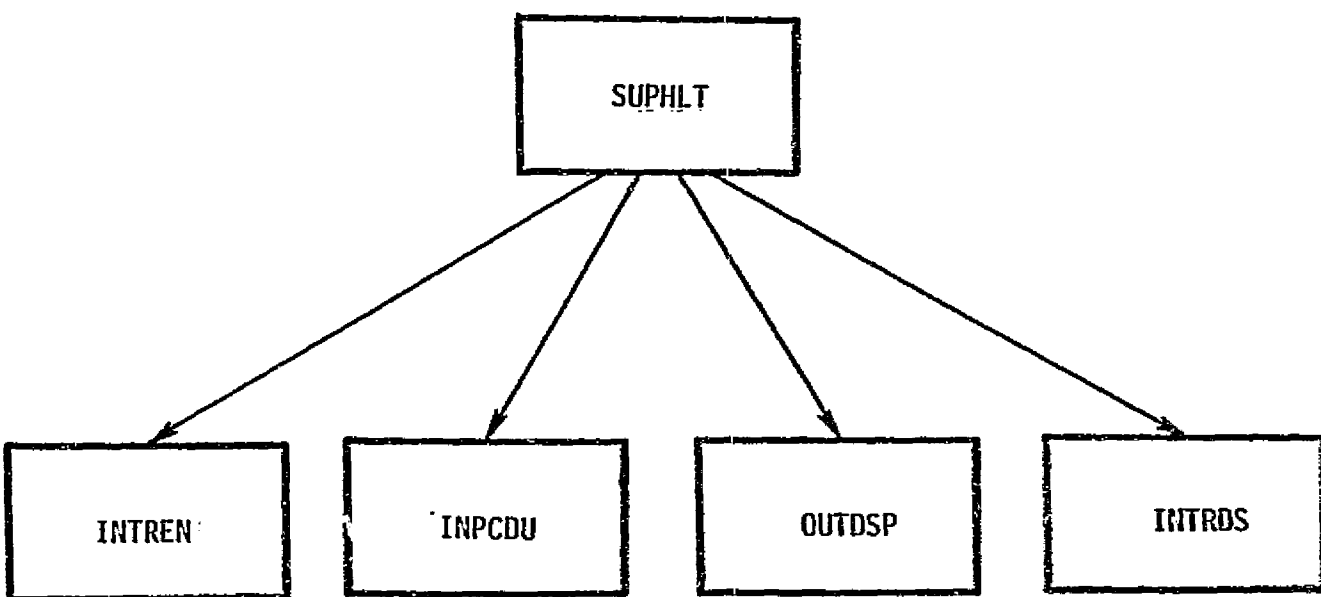
Calls: INTREN, INTRDS, INPCDU, OUTDSP

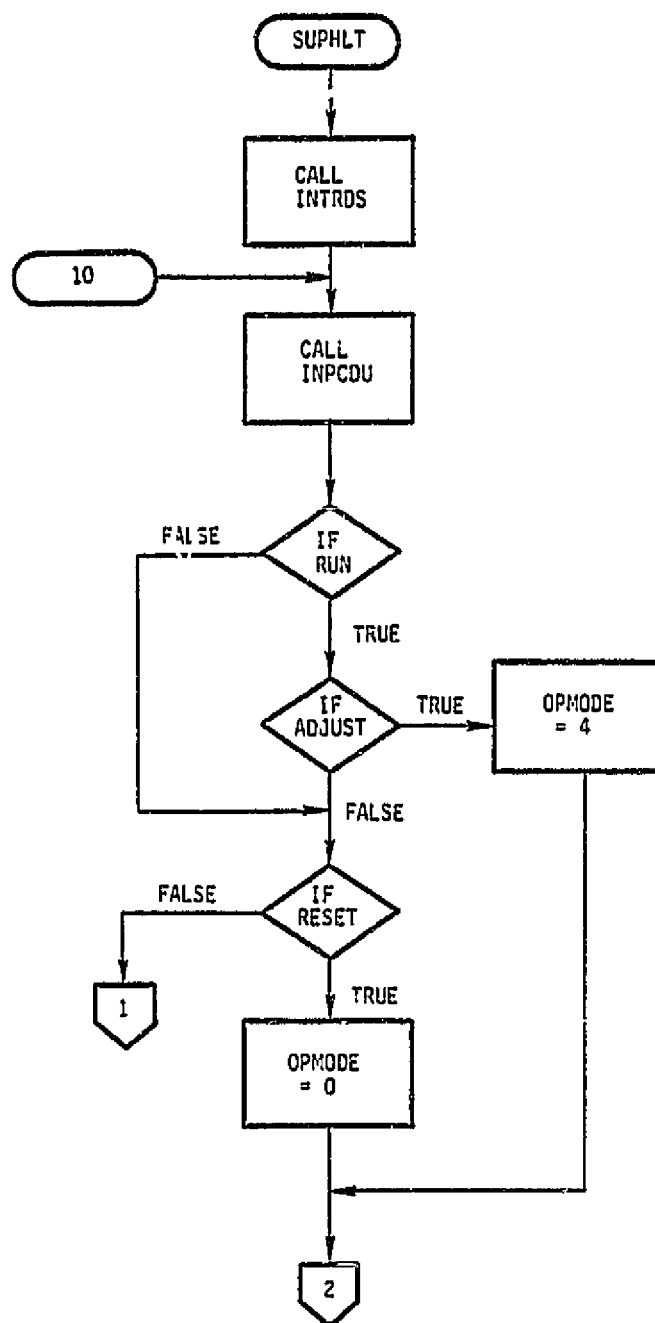
Destroys: DOB

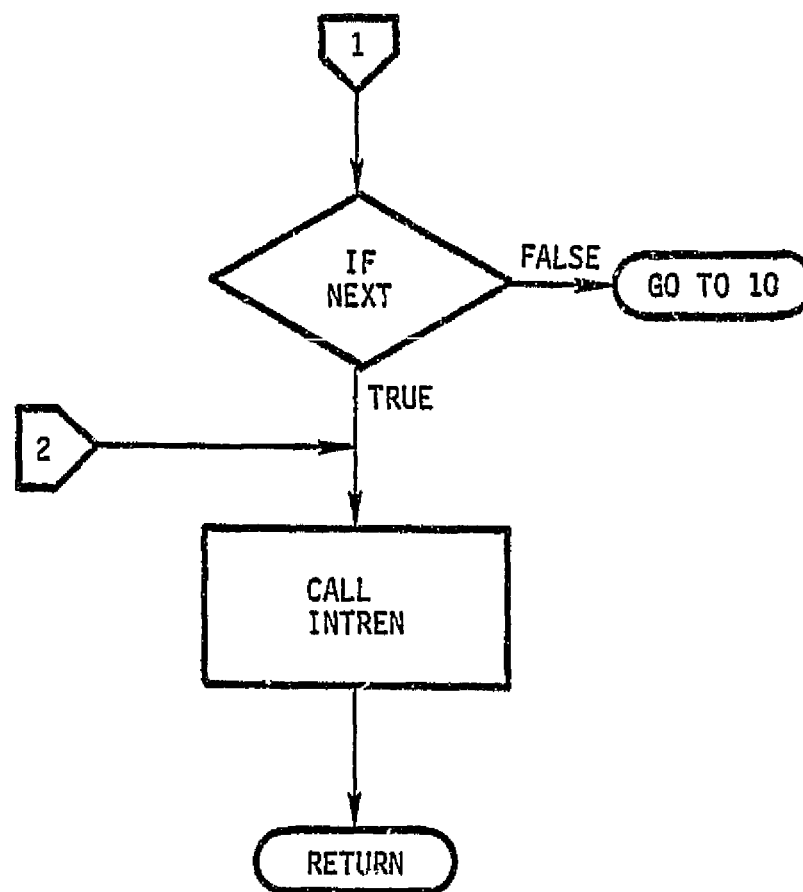
Modifies: OPSVEC

Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish







SUPERVISOR EXECUTION MODULES:

SENINP	-	Sensor input
CONVRT	-	Sensor conversion
CNTOUT	-	Control signal output
ITZINP	-	Initialization routine
RESTRT	-	Reset system, cold start

SUPERVISOR EXECUTION MODULE

Subroutine: SENINP

 SENINP calls sensor input routines under I/O mask control.

Inputs: I/O mask buffer

Outputs: None

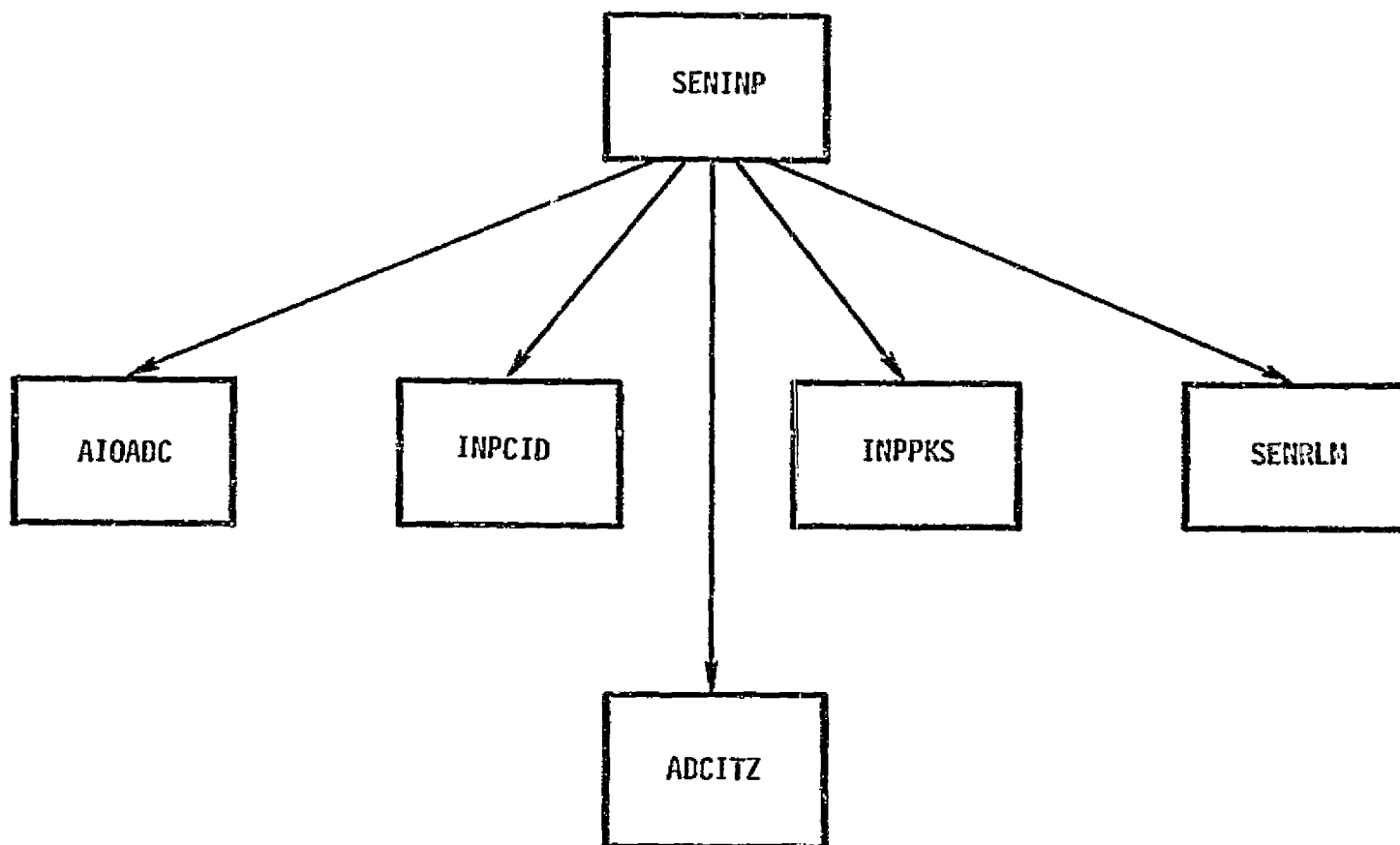
Calls: AIOADC, SENRLM, INPCID, INPPKS, ADCITZ

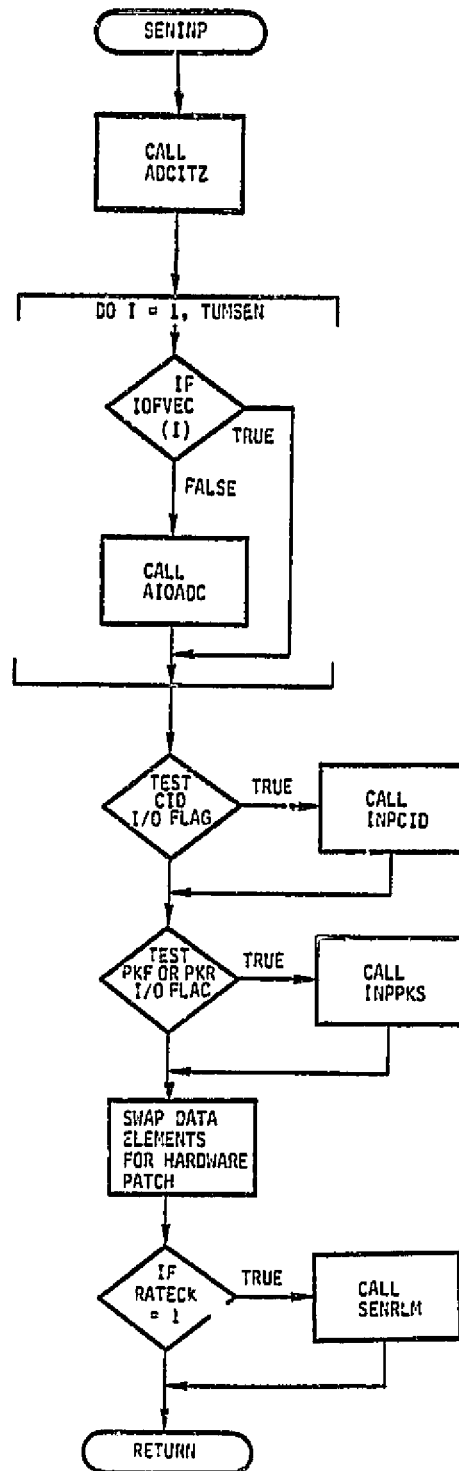
Destroys: Nothing

Modifies: Sensor input buffer

Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish





SUPERVISOR EXECUTION MODULE

Subroutine: CONVRT

CONVRT converts sensor input data to engineering units,
and does range checking.

Inputs: Sensor input data

Outputs: None

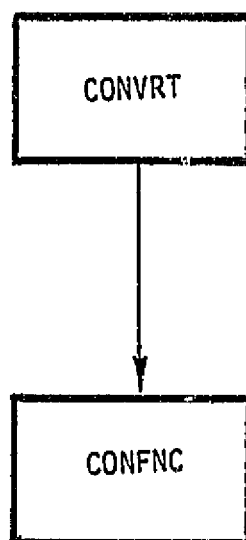
Calls: CONFNC

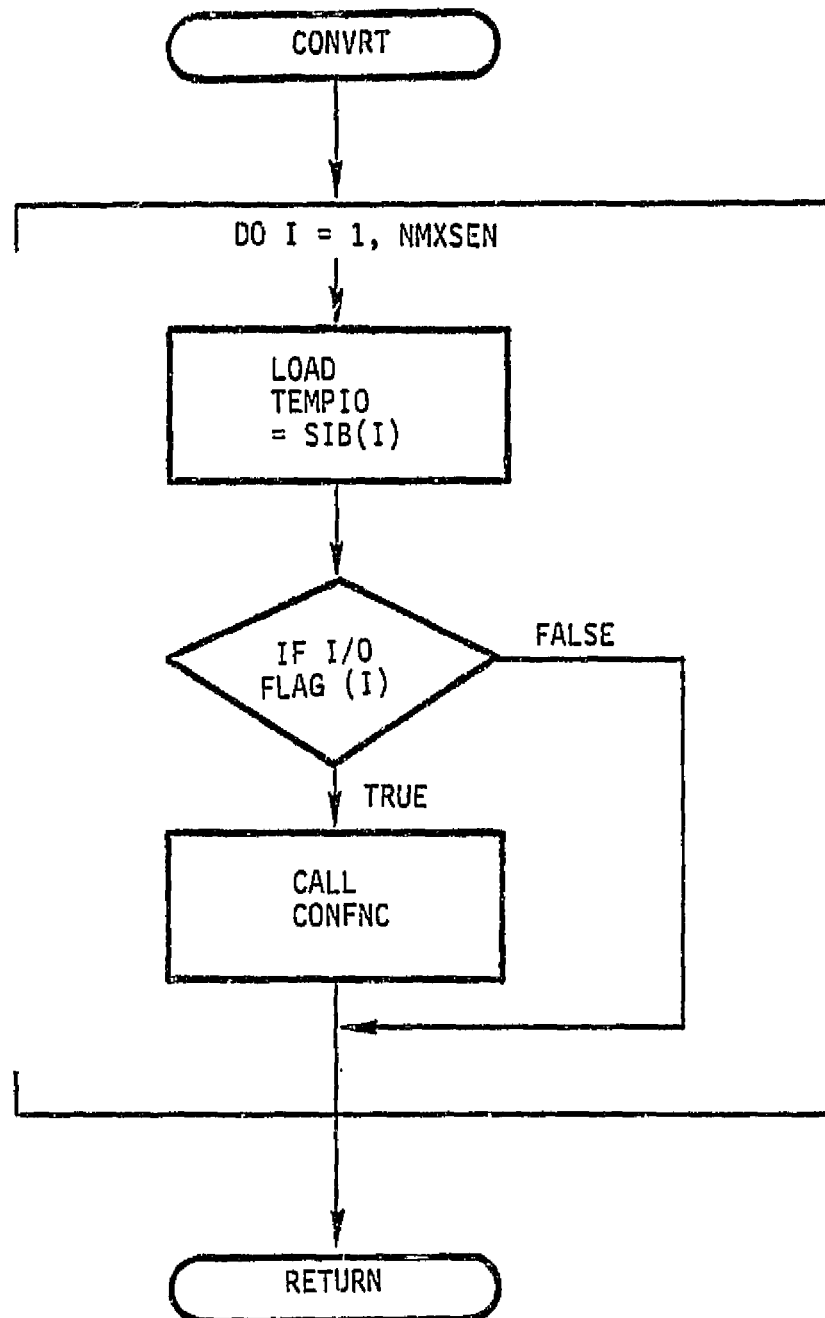
Destroys: Nothing

Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish





SUPERVISOR EXECUTION MODULE

Subroutine: CNTOUT

CNTOUT is the controller output routine. This calls the appropriate output routines as per the I/O controls.

Inputs: I/O control vectors, controller outputs

Outputs: None

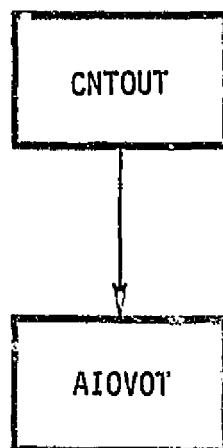
Calls: AIOVOT

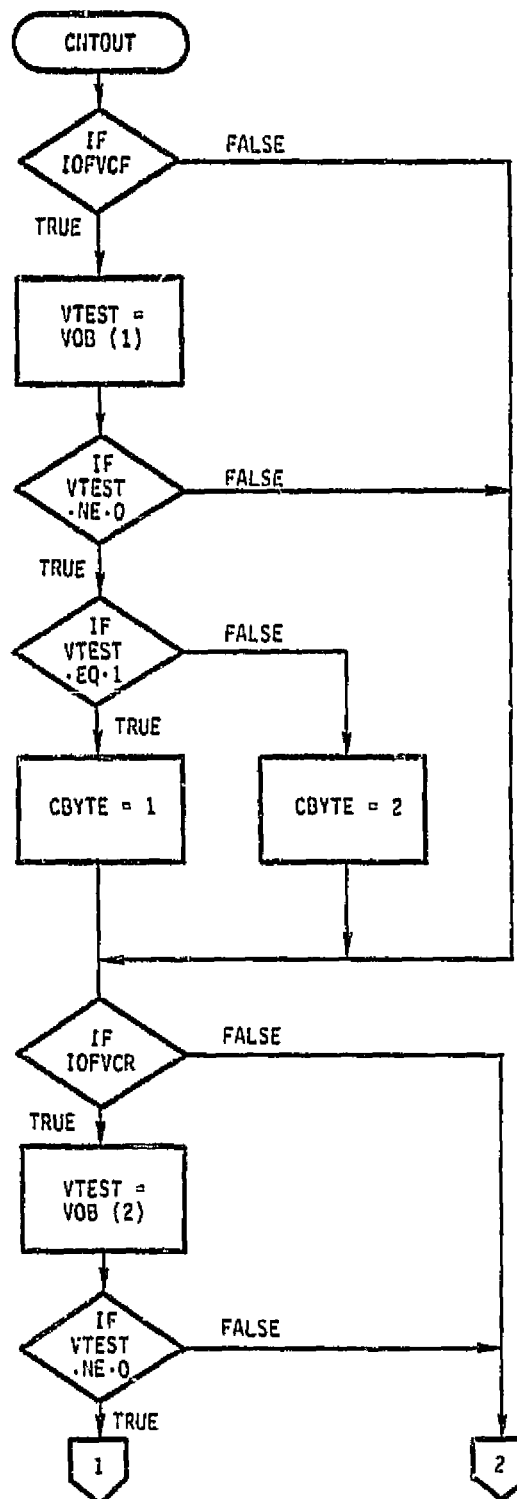
Destroys: Nothing

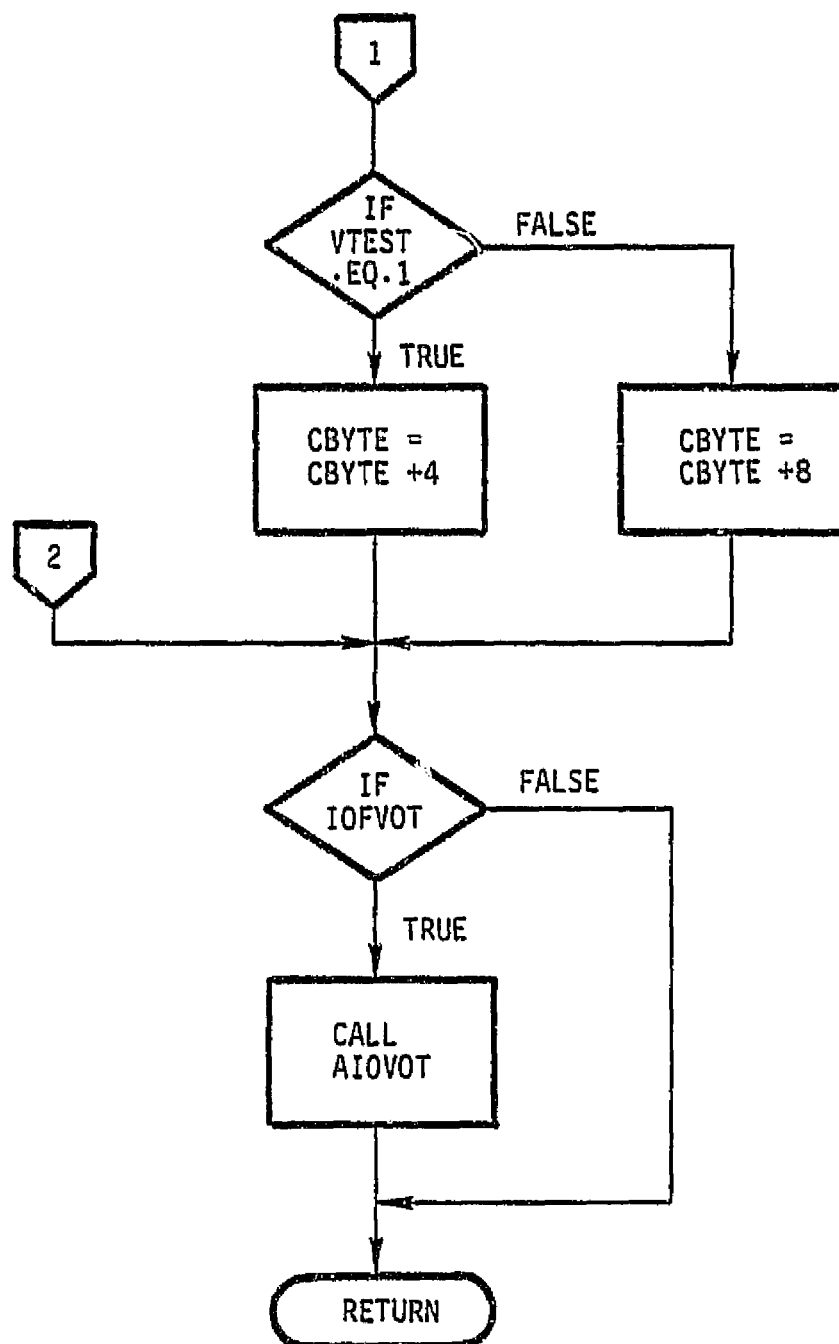
Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish







SUPERVISOR EXECUTION MODULE

Subroutine: ITZINP

ITZINP prompts the user and updates the value of some variables in the program.

Inputs: None

Outputs: None

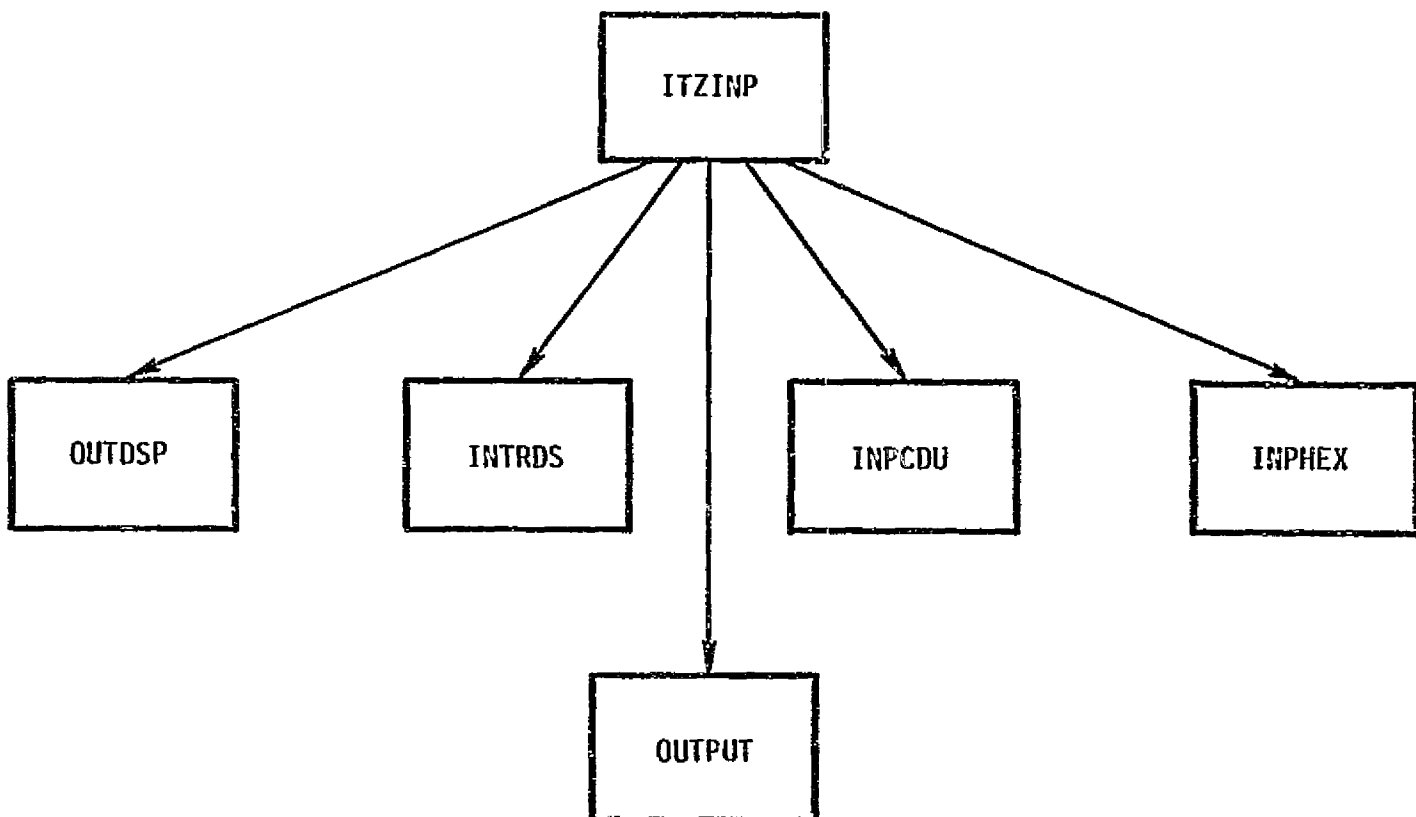
Calls: OUTDSP, INPCDU, INTRDS, INPHEX, OUTPUT

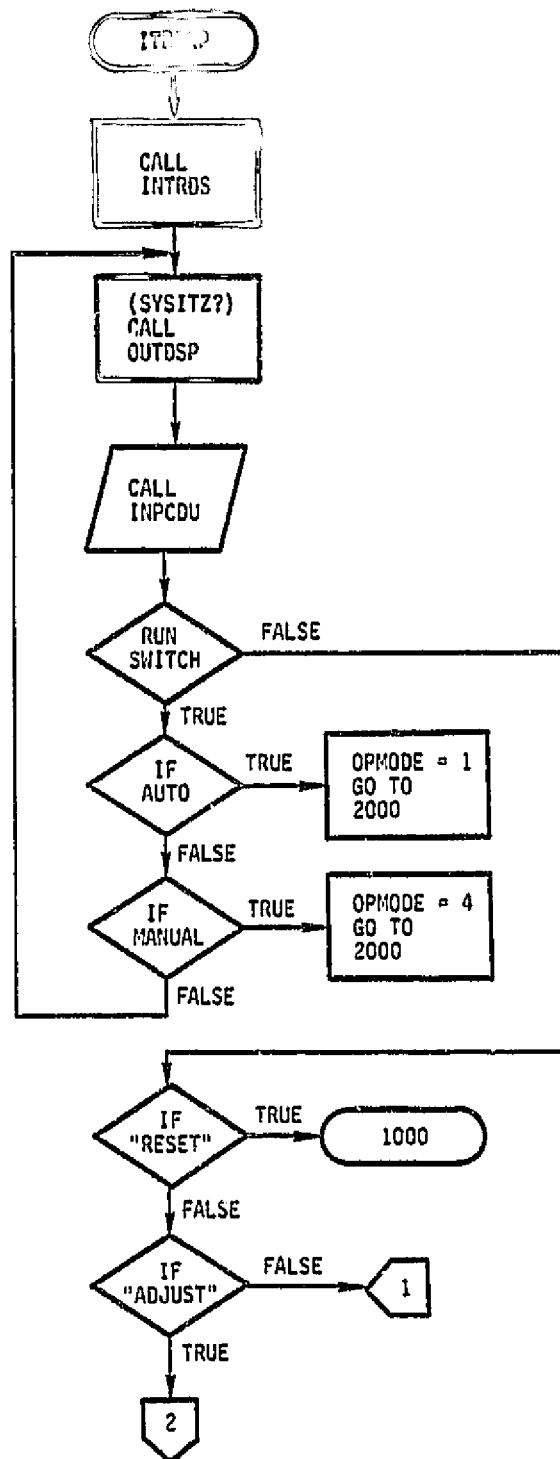
Destroys: CMOS (Optional)

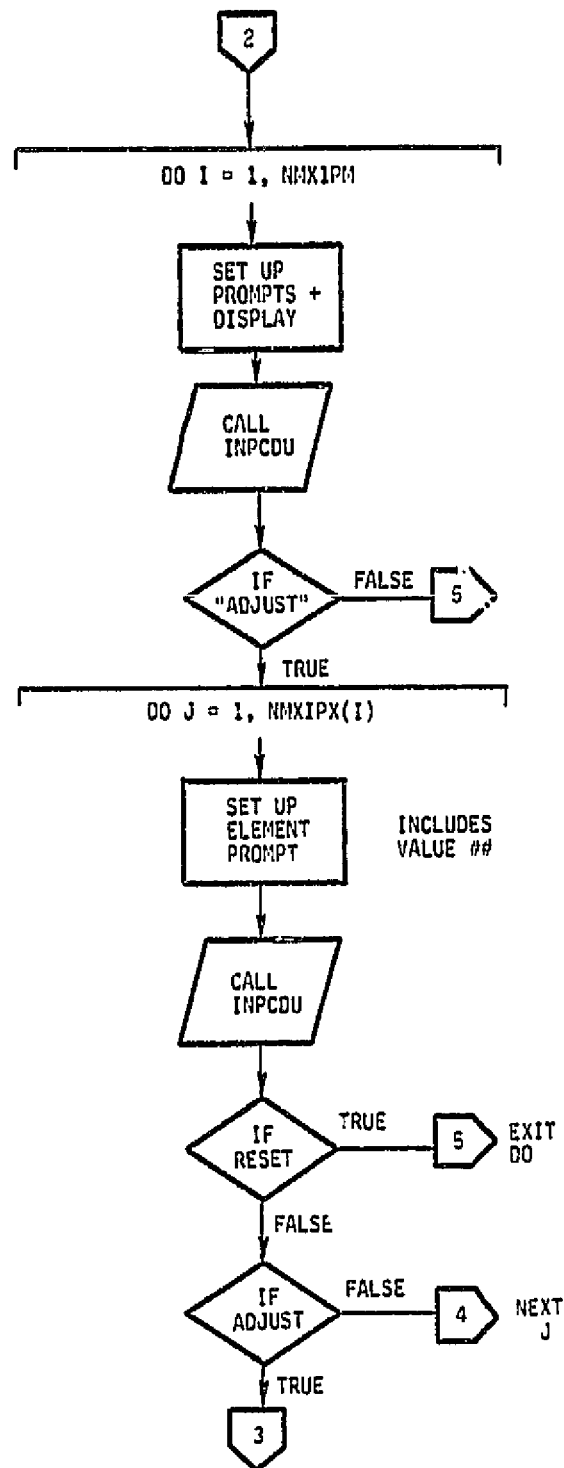
Modifies: Variable input by user

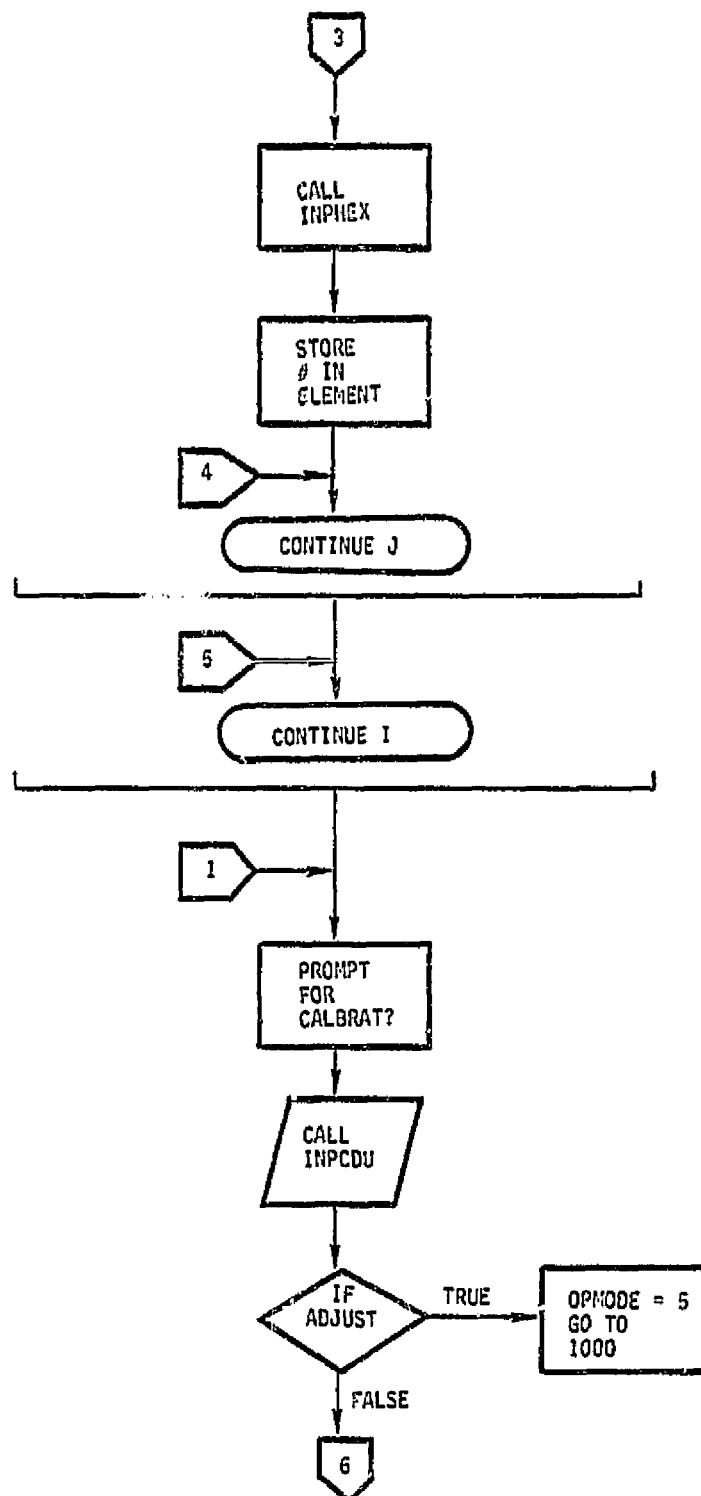
Language: FORTRAN 80

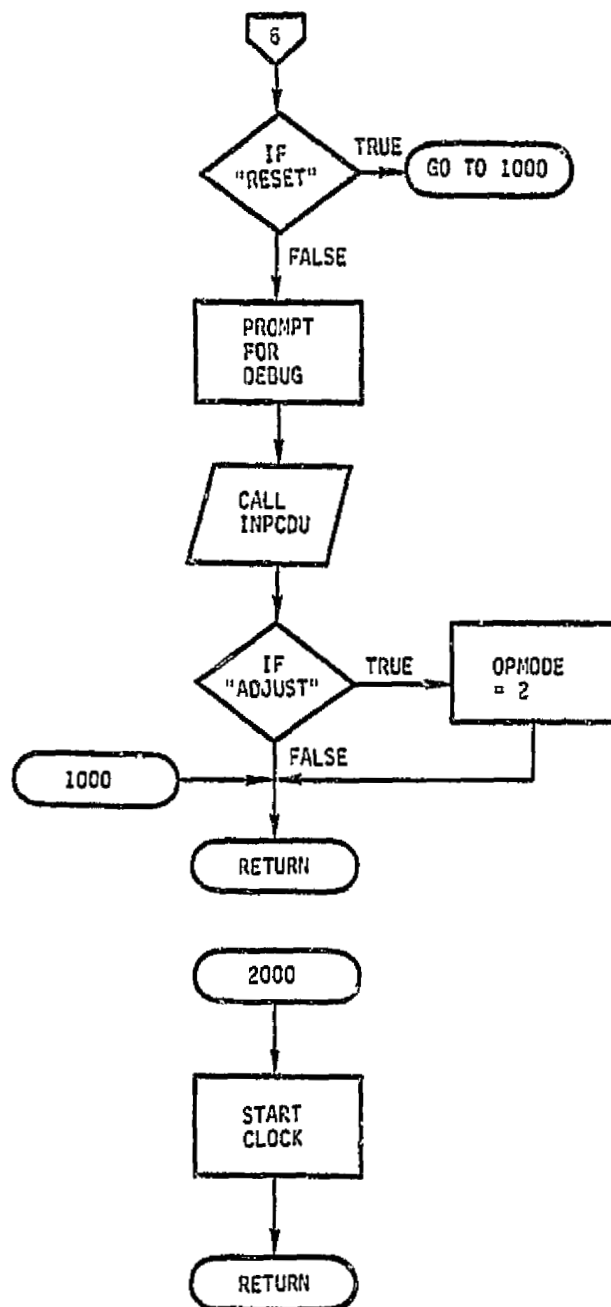
L-V-A: Version 1, Level 1
Roger B. Fish











SUPERVISOR EXECUTION MODULE

Subroutine: RESTRT

RESTRT initializes and clears data which is run dependent
for restart.

Inputs: None

Outputs: None

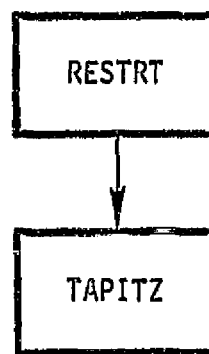
Calls: TAPITZ

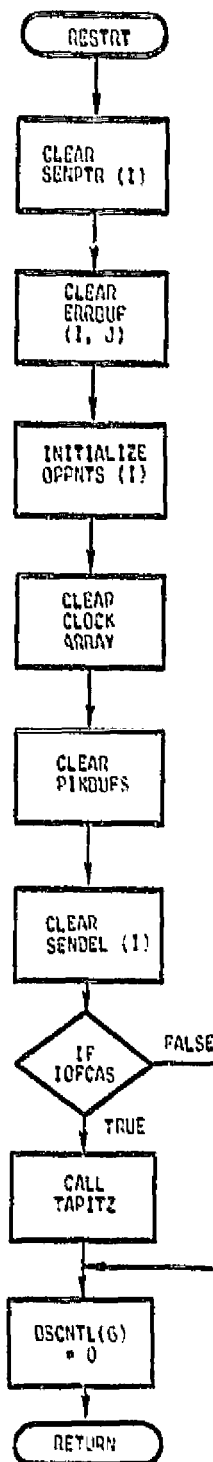
Destroys: SENPTR(I), ERRBUF(I), TFLAG(I), PKBUFS,
SENDEL(I)

Modifies: Status, pointers, OPMODE

Language: FORTRAN

L-V-A: Version 1, Level 1
Roger B. Fish





SUPERVISOR EXECUTION MODULE:

Control Routines

CNTRFD - Front drum control

CNTRRD - Rear drum control

SUPERVISOR EXECUTION MODULE

Subroutine: CNTRFD

 CNTRFD computes the front drum control signal. Algorithm
 is controlled by inputs and mode control.

Inputs: Sensor data, control information

Outputs: Front drum control signal

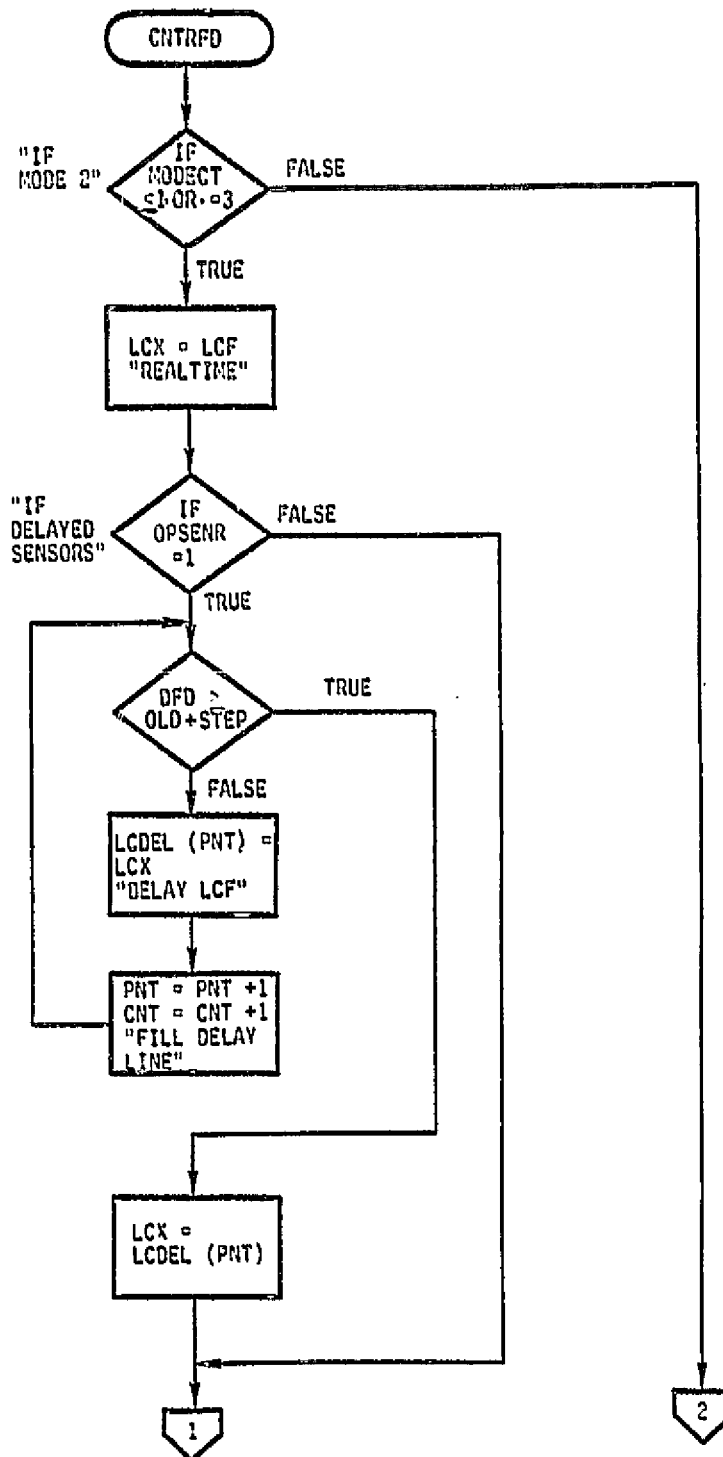
Calls: Nothing

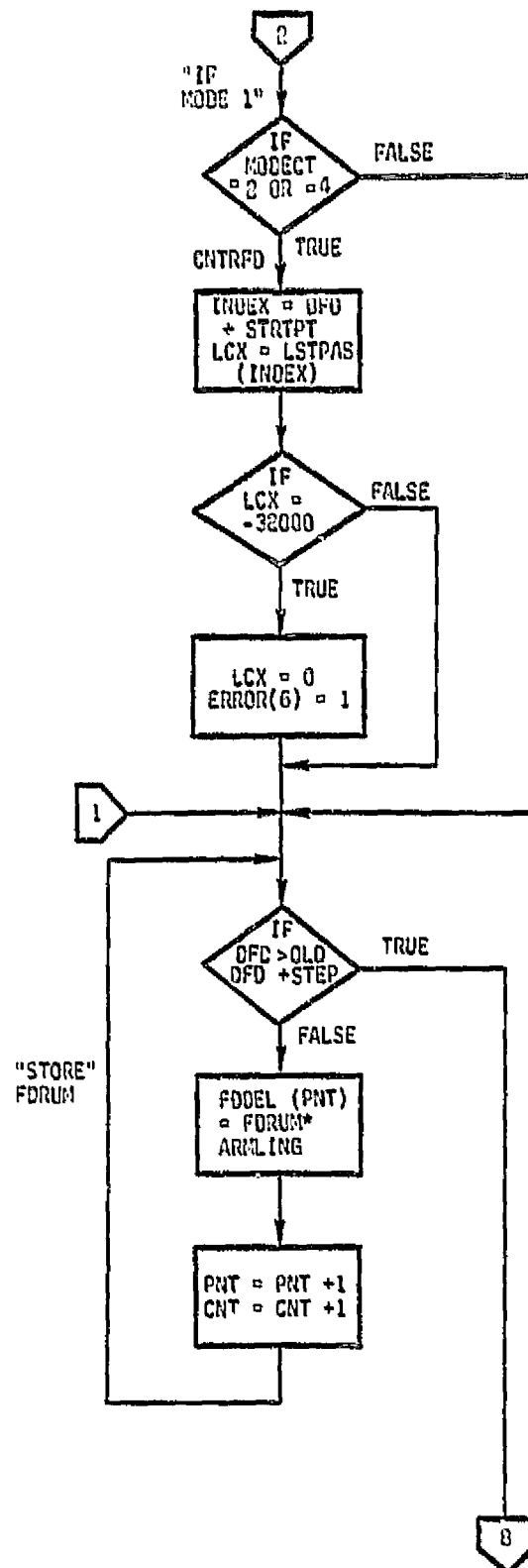
Destroys: Nothing

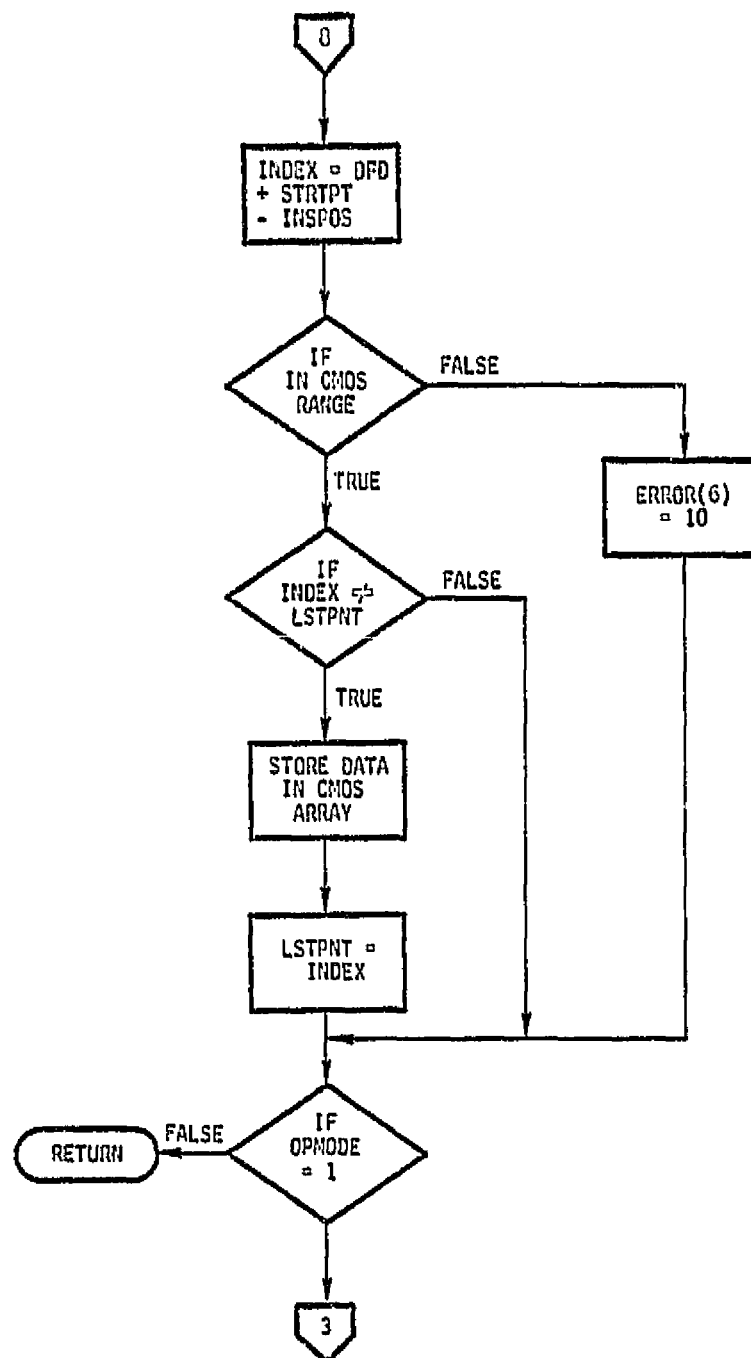
Modifies: Control output, front (VOB)

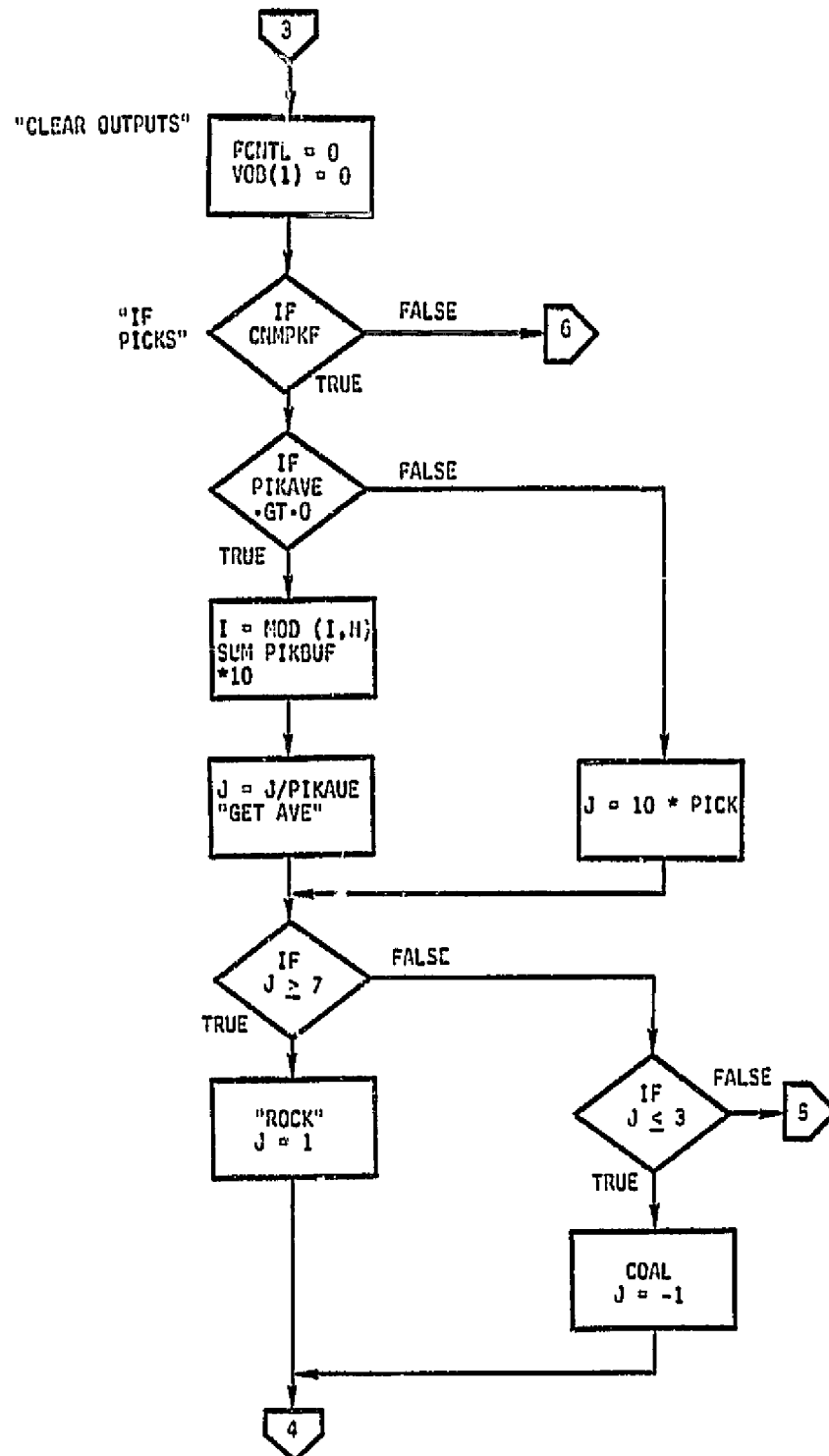
Language: FORTRAN 80

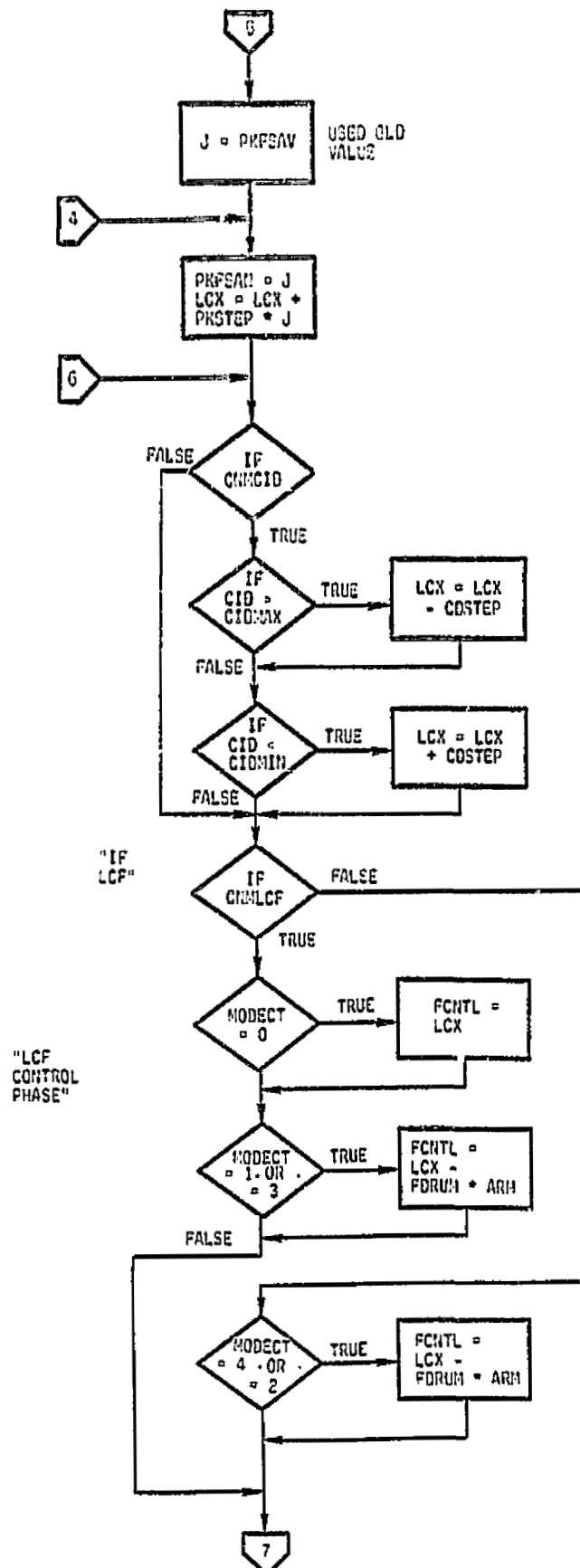
L-V-A: Version 1, Level 1
 Roger B. Fish

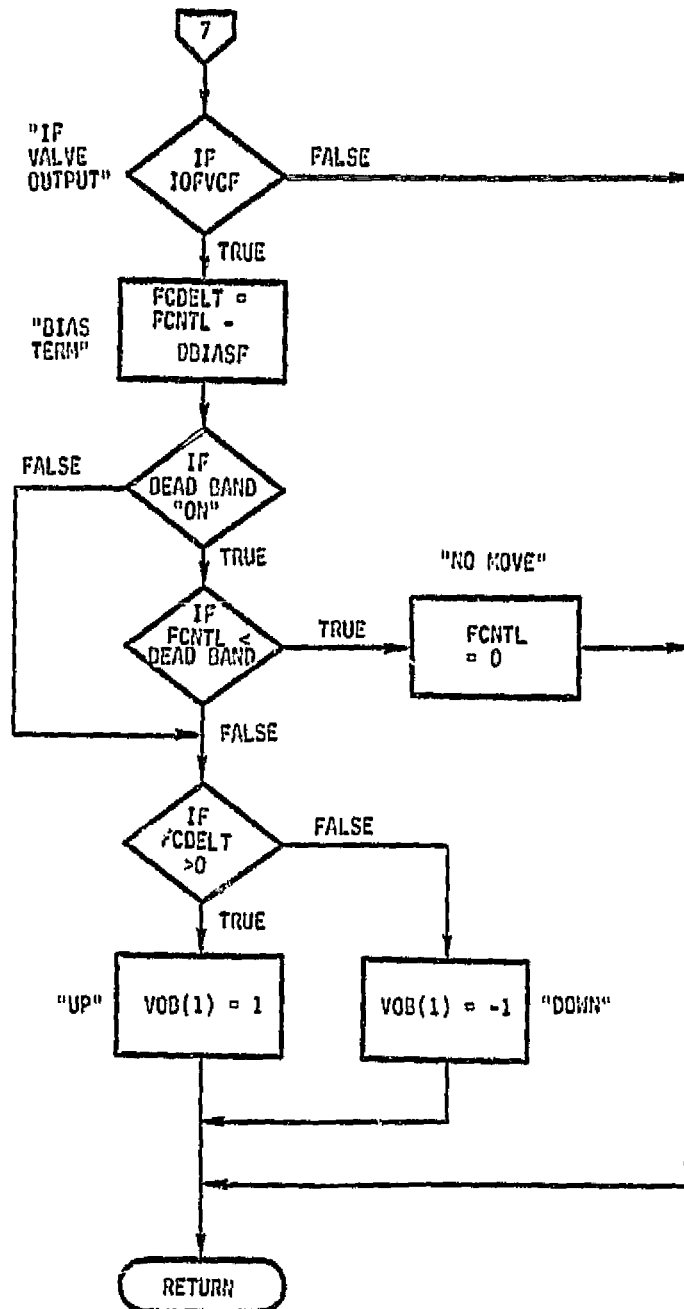












SUPERVISOR EXECUTION MODULE

Subroutine: CNTRRD

 CNTRRD computes the rear drum control signal. Algorithm
 is controlled by inputs and mode control.

Inputs: Sensor data, control information

Outputs: Front drum control signal

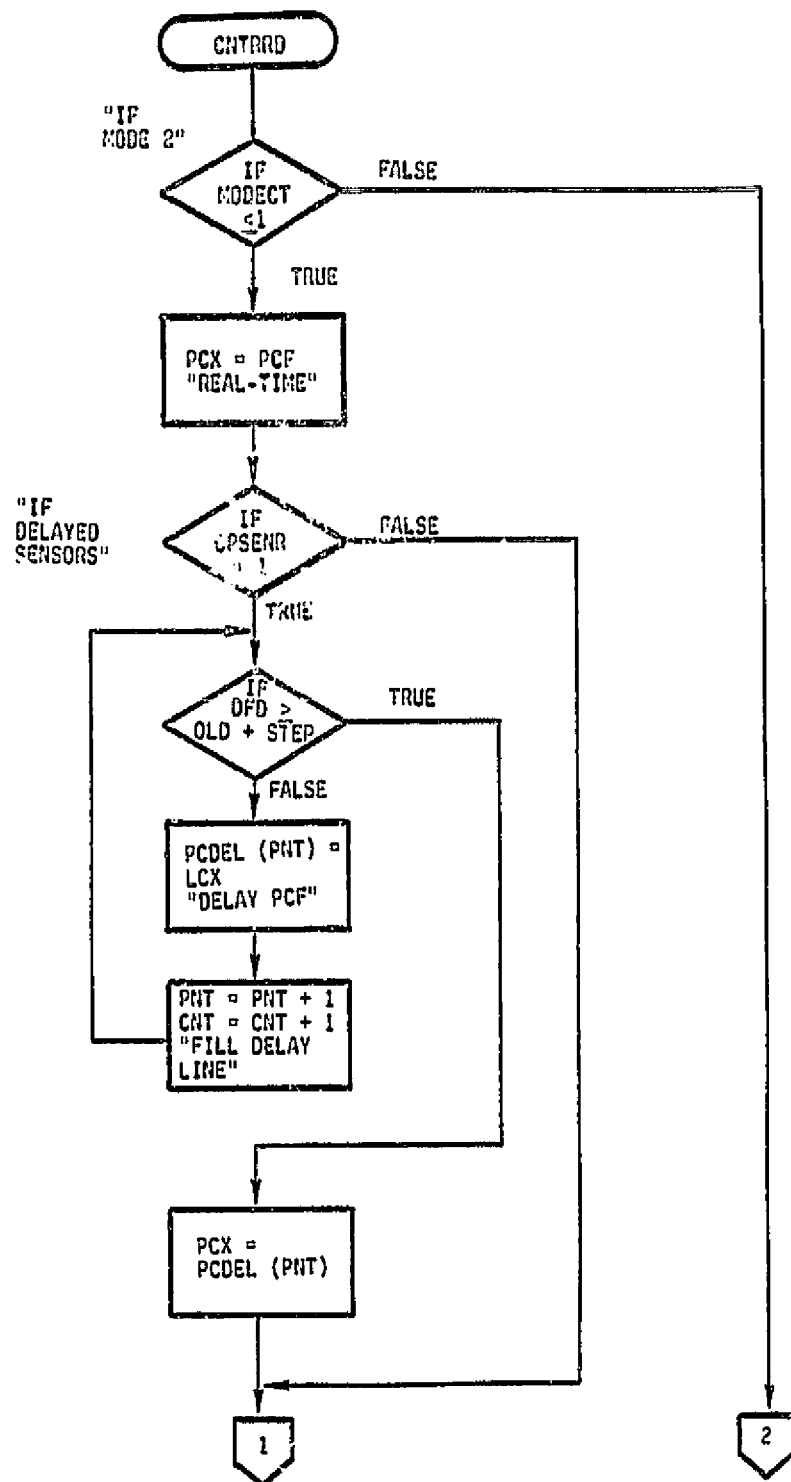
Calls: Nothing

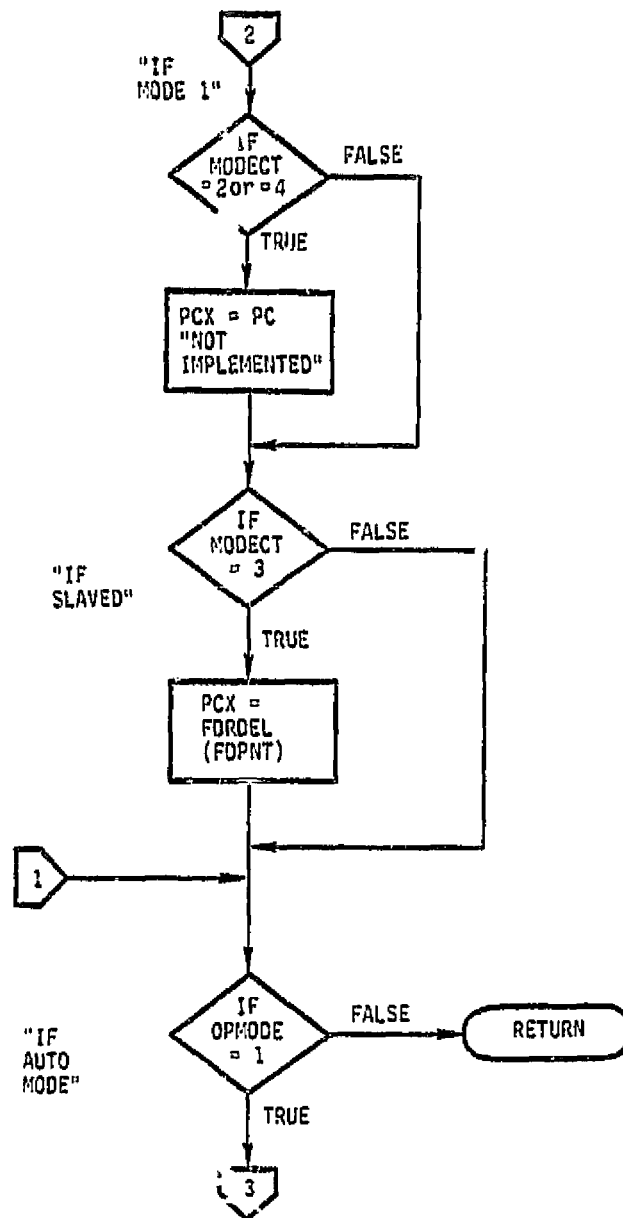
Destroys: Nothing

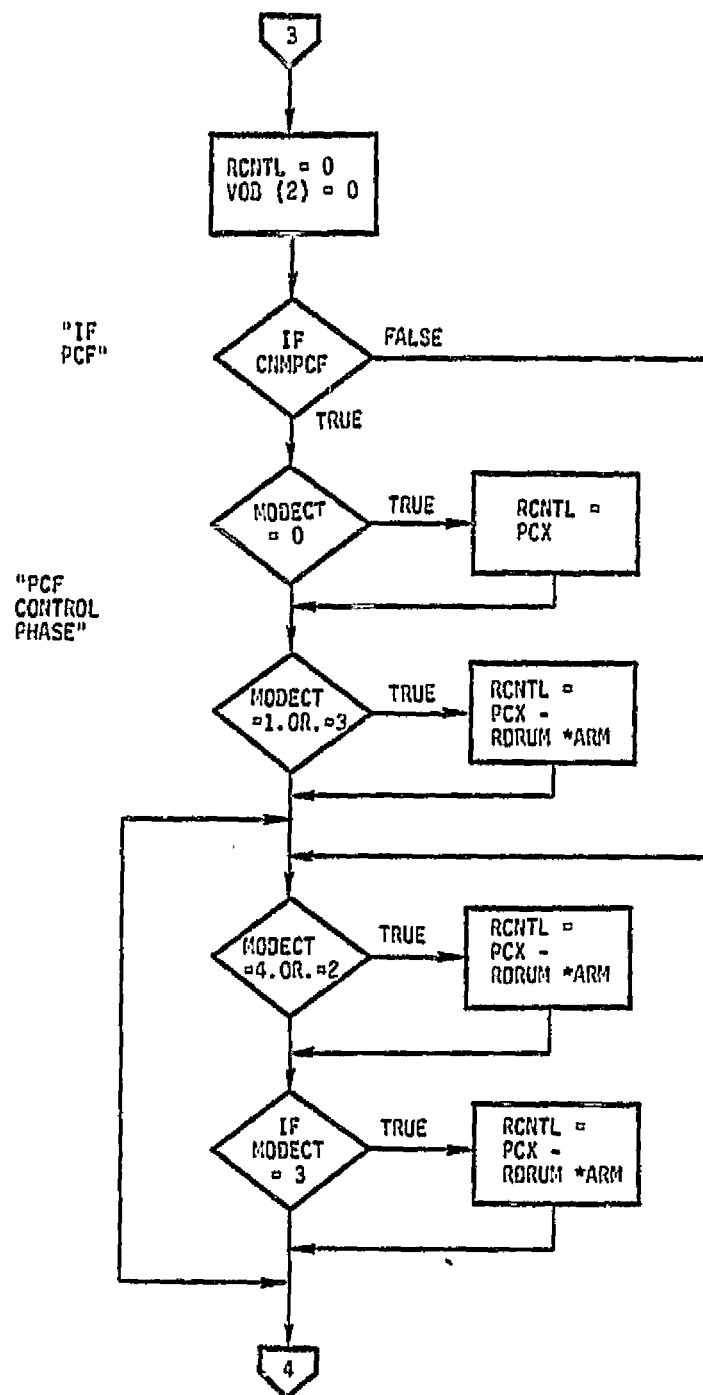
Modifies: Control output, rear (VOB)

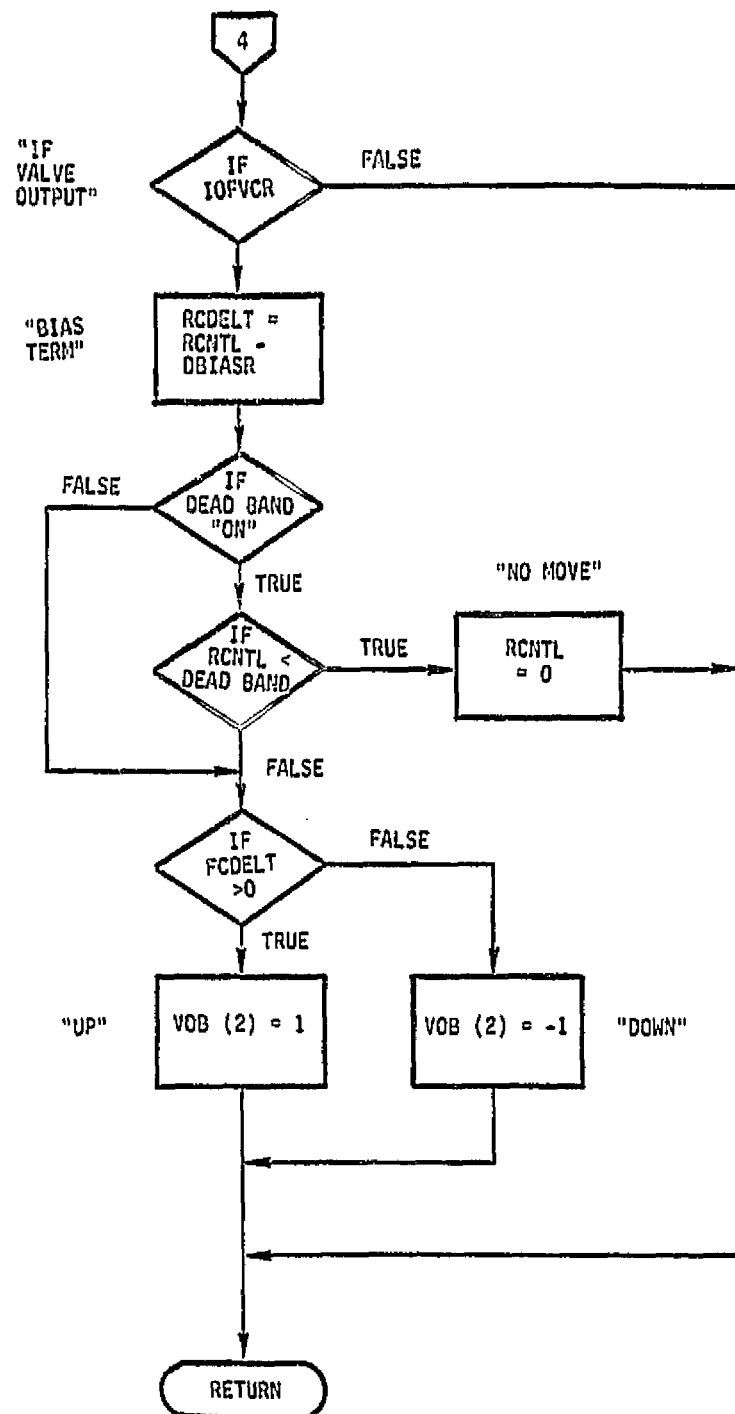
Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish









SUPERVISOR EXECUTION MODULES:

Debugger Routines

DEBUGCI	-	Input control
DEBUGCO	-	Output
DEBUGIN	-	Sensor scan
DEBUGOT	-	Output (valve)
DEBUGIO	-	I/O channel test
DEBUGTP	-	Tape check
DEBUGCK	-	Clock test
DEBUGME	-	Memory display/modify

SUPERVISOR EXECUTION MODULE

Subroutine: DEUGCI

DEUGCI reads debug control information from control/
display panel.

Inputs: Debug control vector

Outputs: None

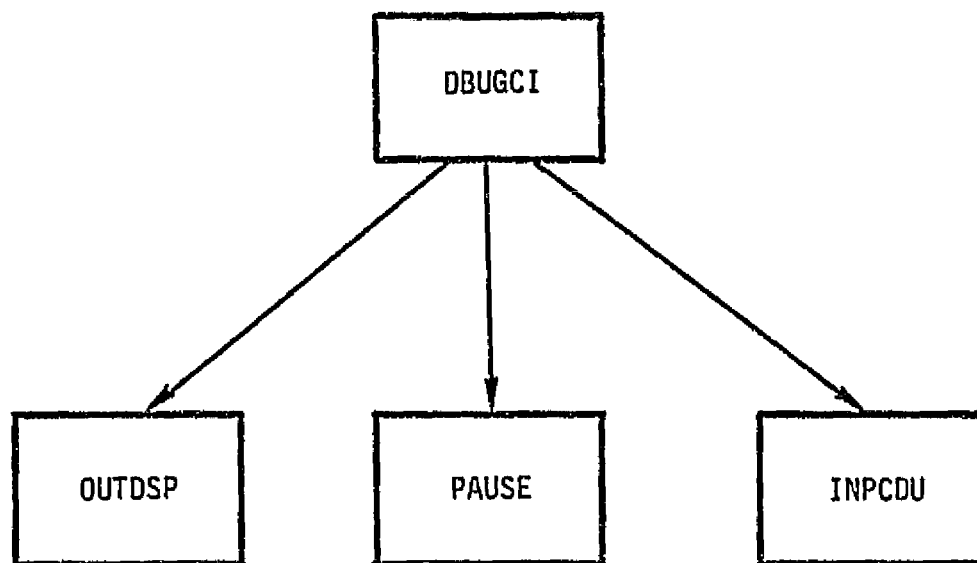
Calls: INPCDU, OUTDSP, PAUSE

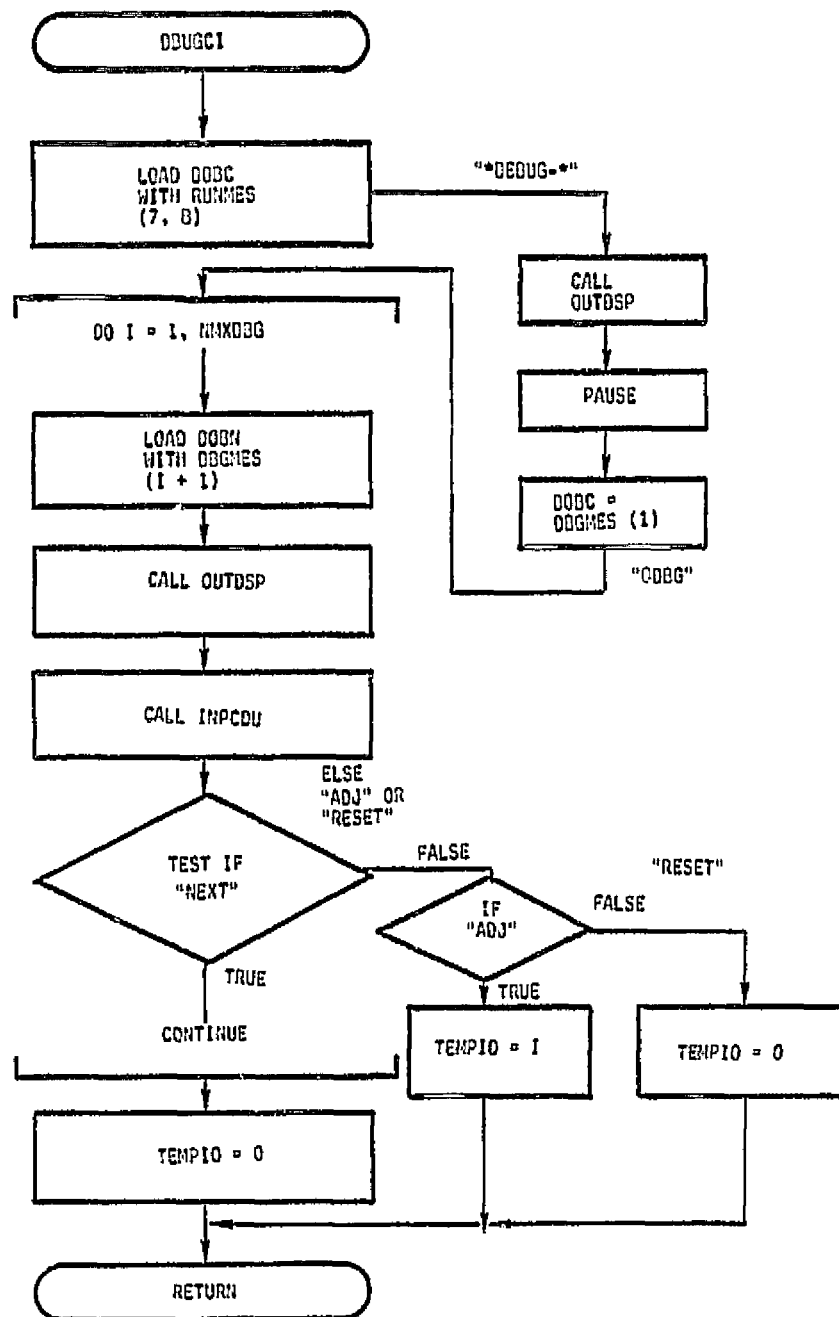
Destroys: Nothing

Modifies: Debug control vector

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





SUPERVISOR EXECUTION MODULE

Subroutine: DEBUGCO

DEBUGCO outputs debug information to control/display panel.

Inputs: Debug control vector

Outputs: Debug messages

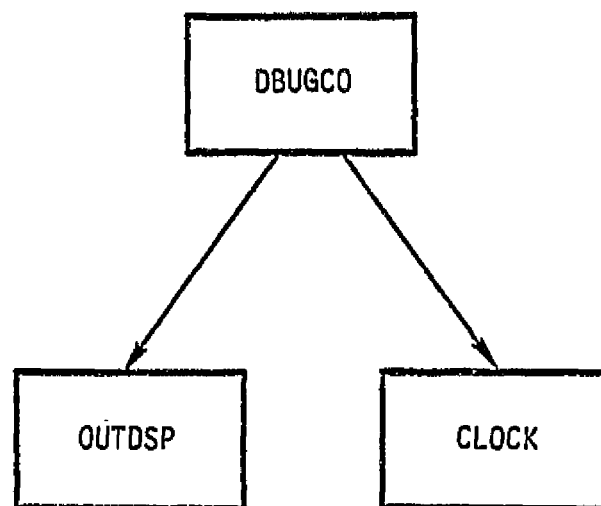
Calls: OUTDSP, CLOCK

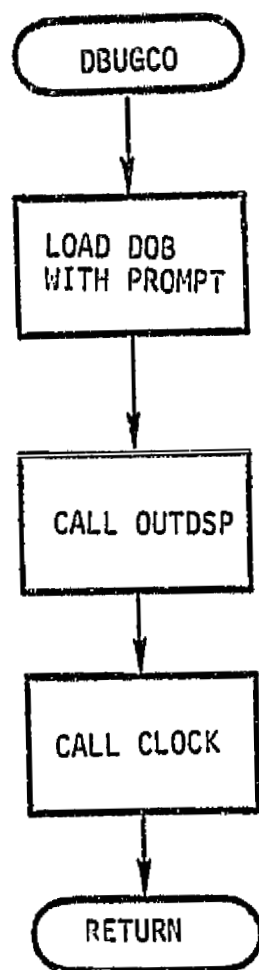
Destroys: Nothing

Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





"@DBG"

SUPERVISOR EXECUTION MODULE

Subroutine: DEBUGIN (FLAG)

DEBUGIN reads sensor data via I/O routines as per input.

Inputs: Sensor buffer, control vector

Outputs: None

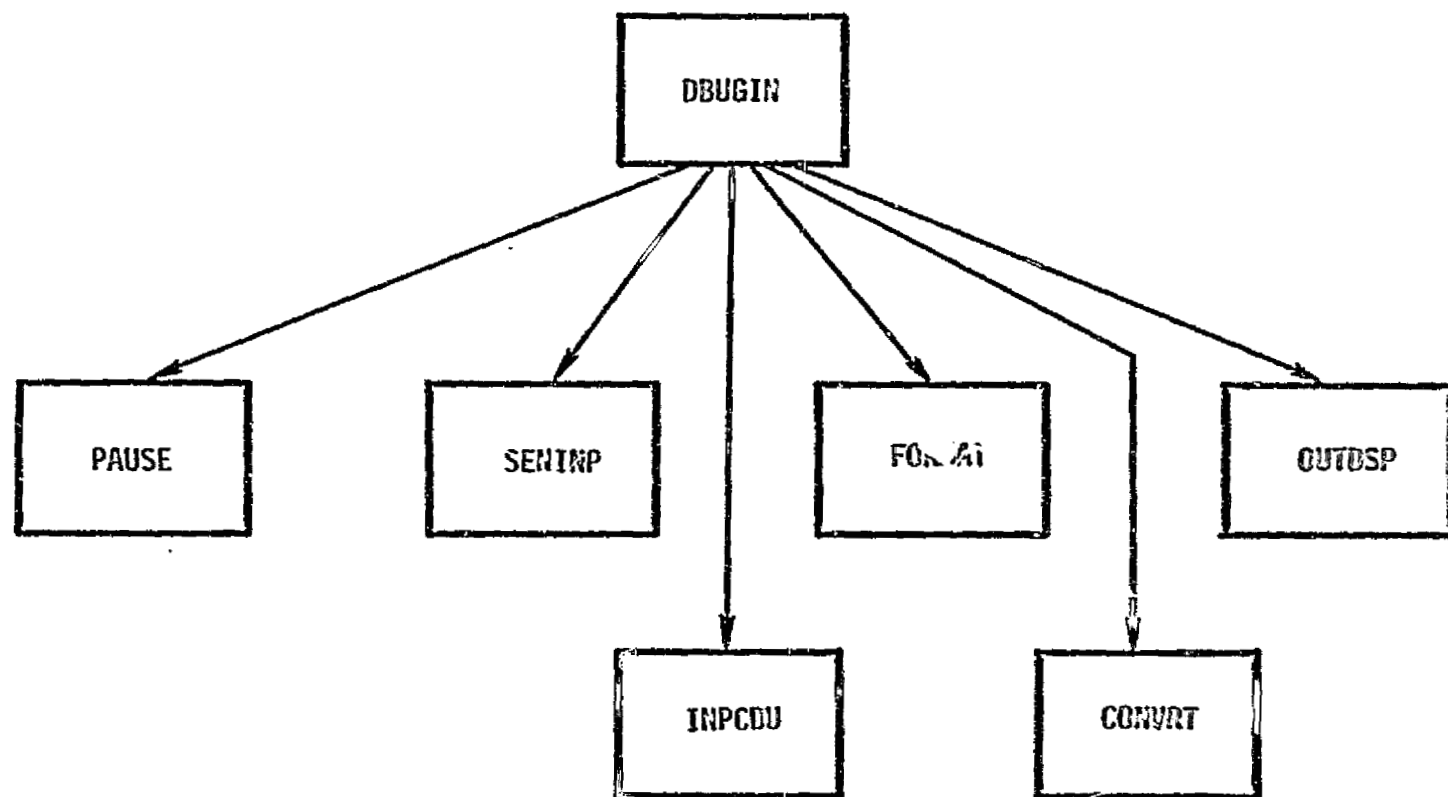
Calls: SENINP, PAUSE, OUTDSP, FORMAT, INPCDU,
CONVRT

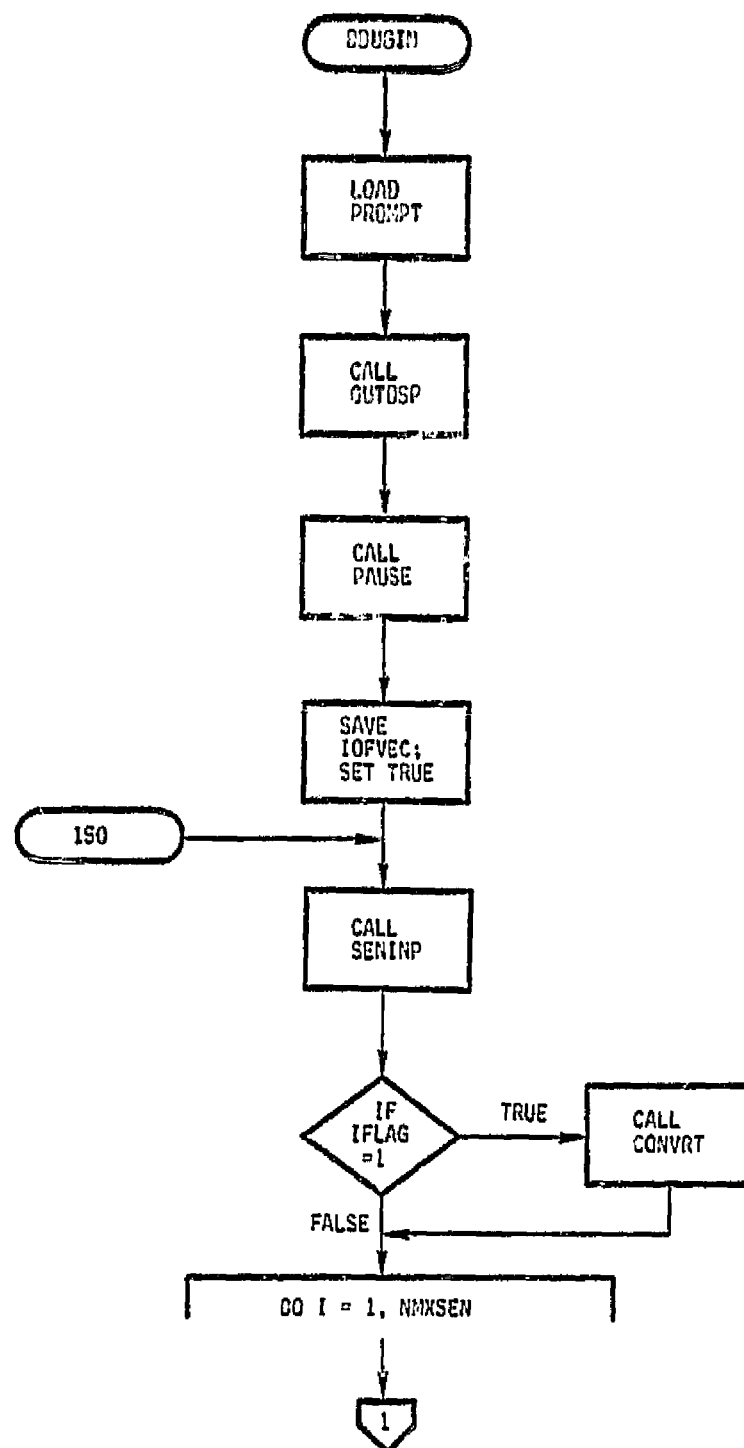
Destroys: Nothing

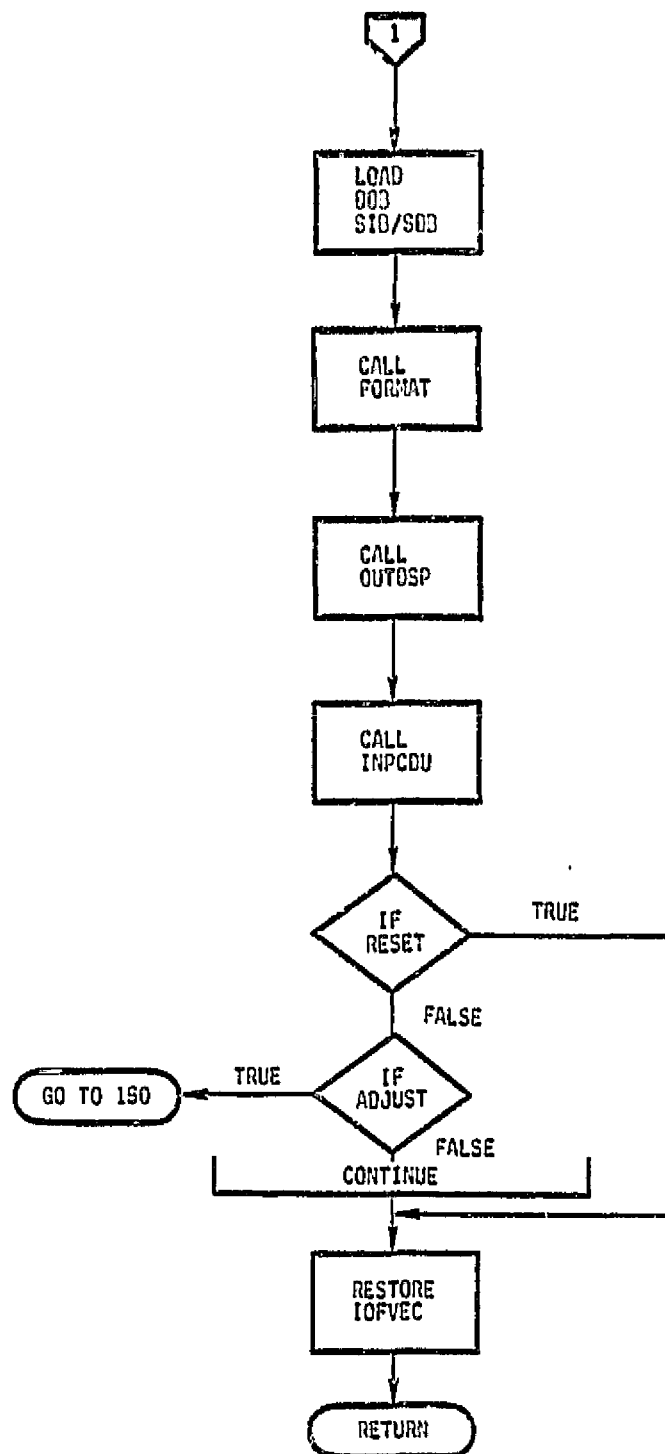
Modifies: Sensor input data (SDB)

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish







SUPERVISOR EXECUTION MODULE

Subroutine: DEUGOT

DEUGOT outputs debug control signals to front and rear arms.

Inputs: Debug control vector

Outputs: Drum control signals

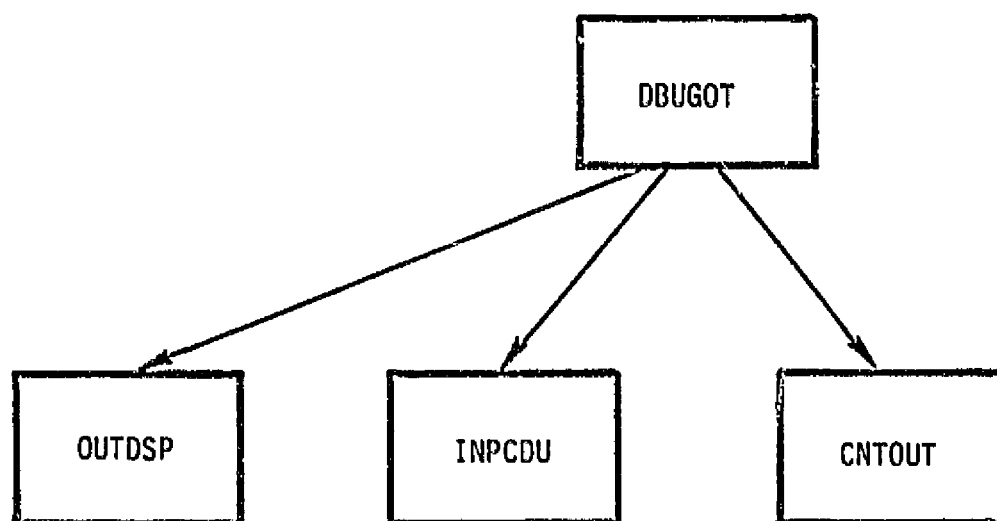
Calls: INPCDU, CNTOUT, OUTDSP

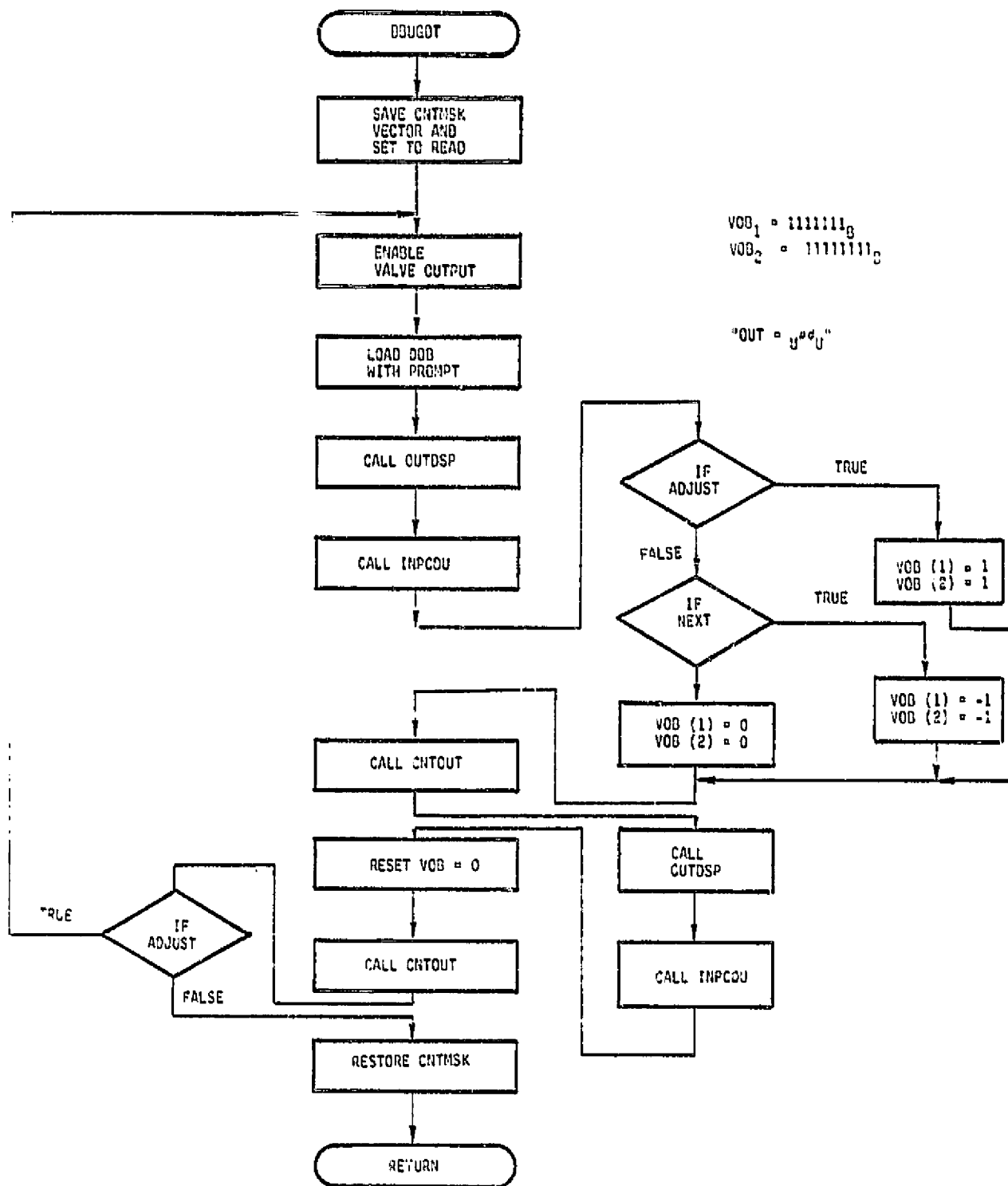
Destroys: Nothing

Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





SUPERVISOR EXECUTION MODULE

Subroutine: DEBUGIO

DEBUGIO reads an I/O channel or port and displays value.

Inputs: None

Outputs: None

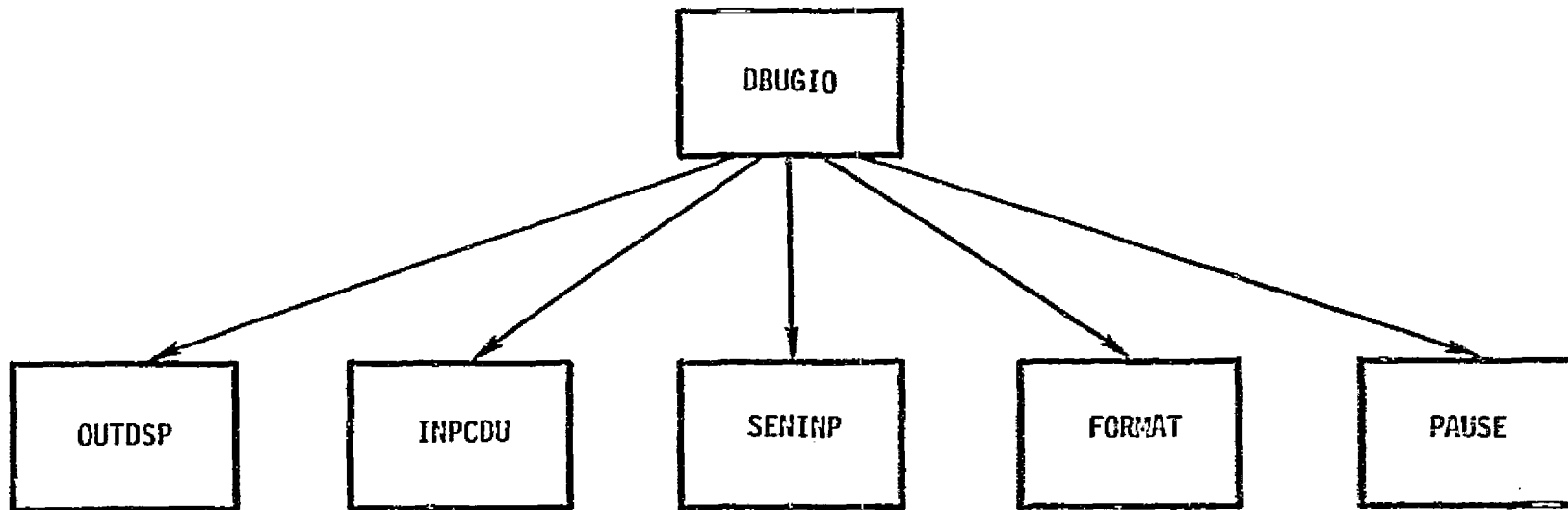
Calls: FORMAT, OUTDSP, PAUSE, INPCDU, SENINP

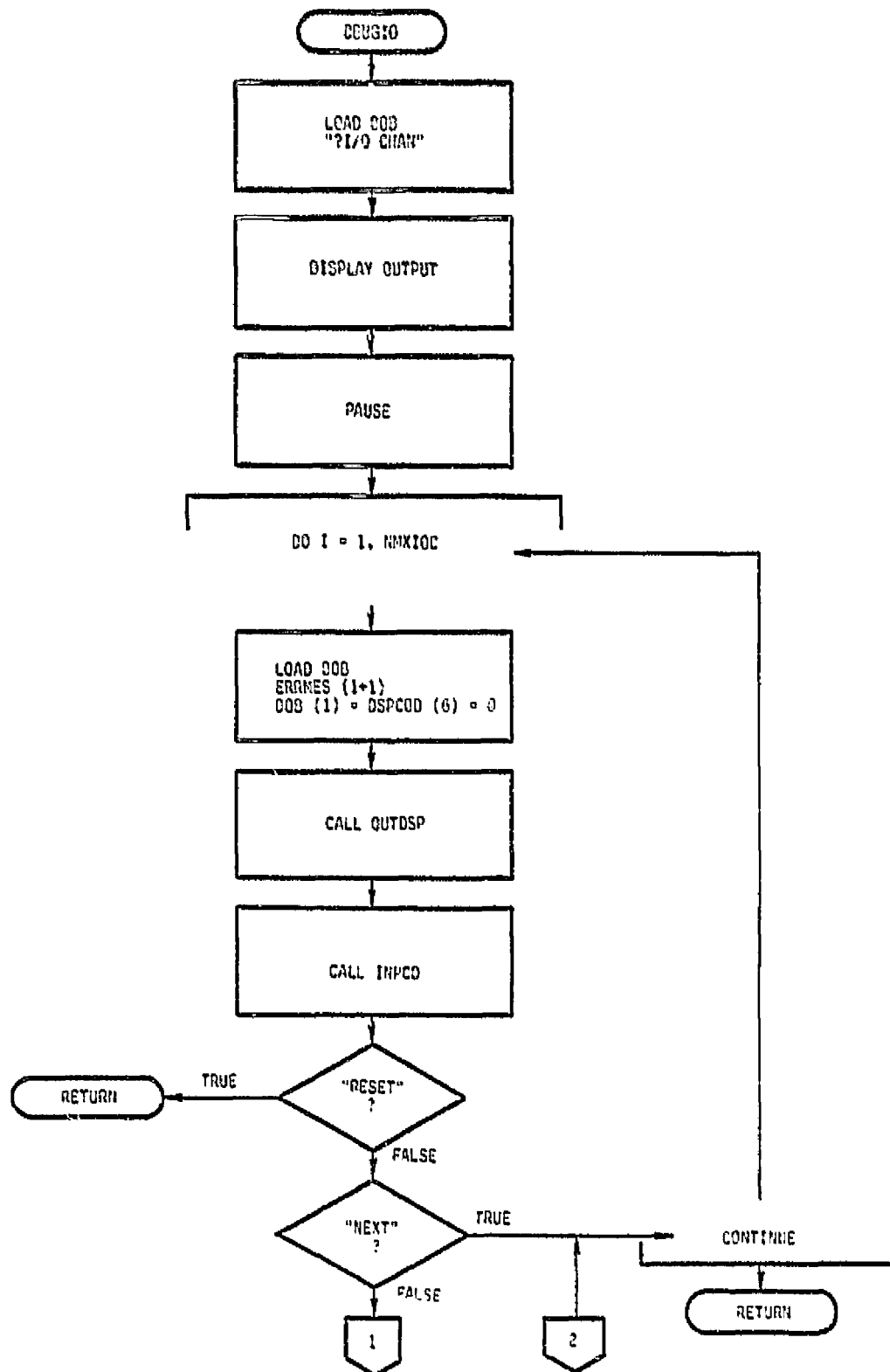
Destroys: Nothing

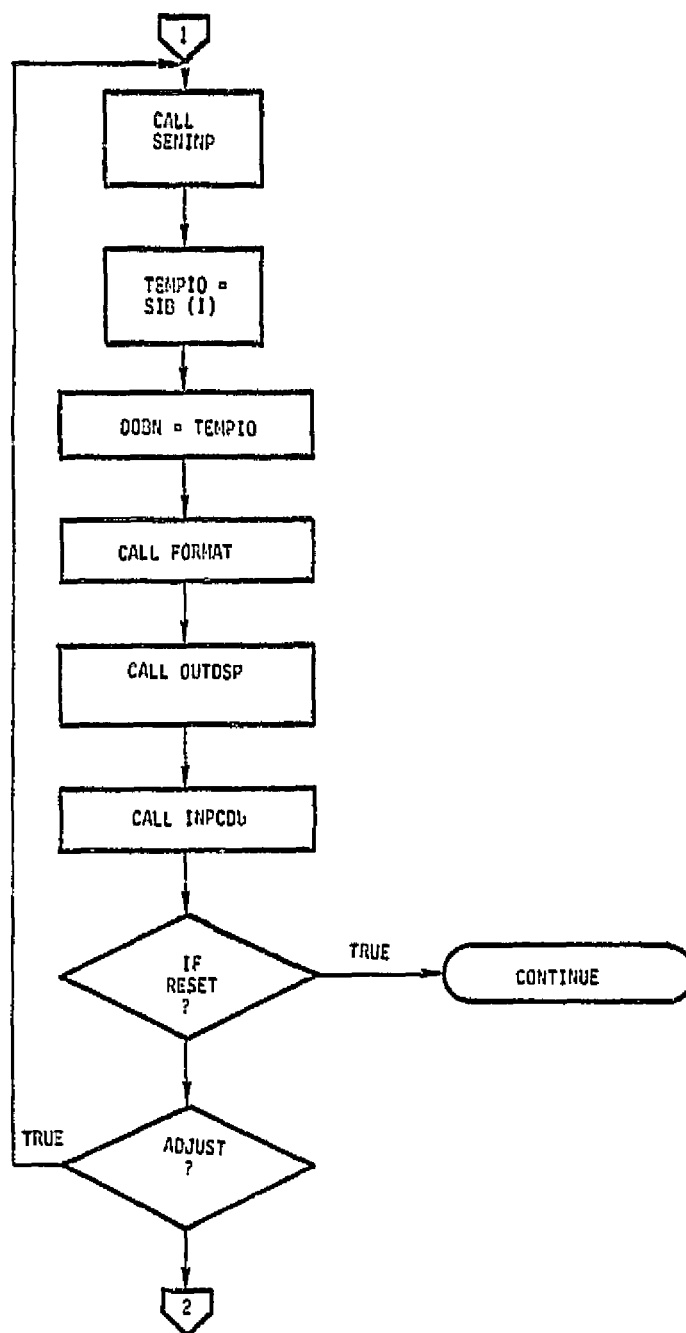
Modifies: DOB

Language: FORTRAN

L-V-A: Version 1, Level 1
 Roger B. Fish







SUPERVISOR EXECUTION MODULE

Subroutine: DBUGTP

 DBUGTP checks tape cassette operation.

Inputs: None

Outputs: None

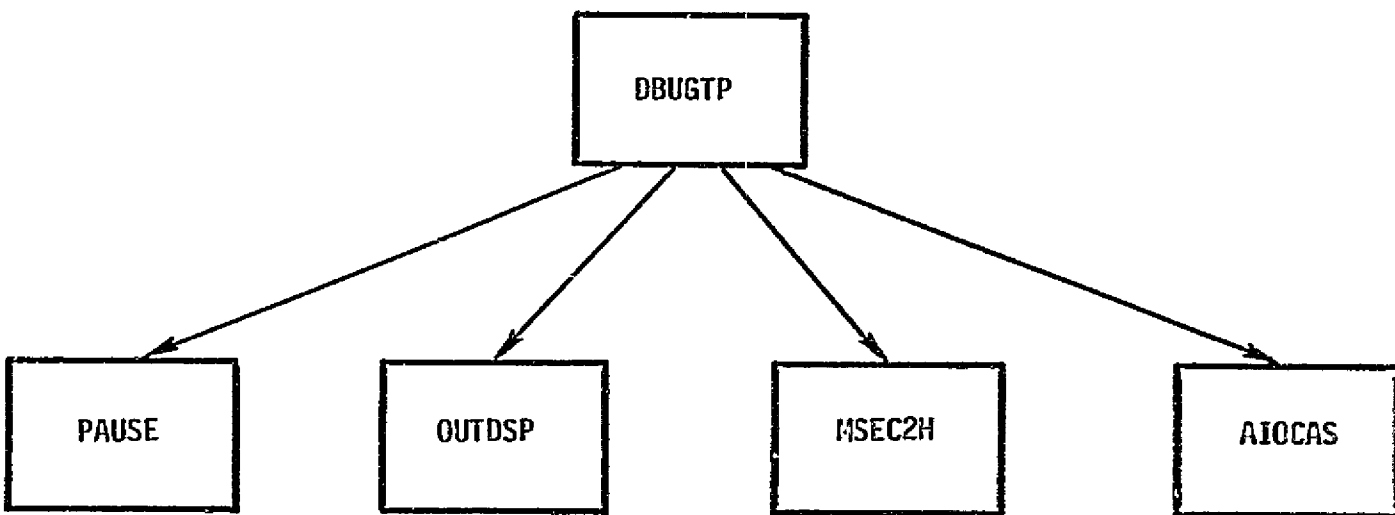
Calls: AIOCAS, MSEC2H, PAUSE, OUTDSP

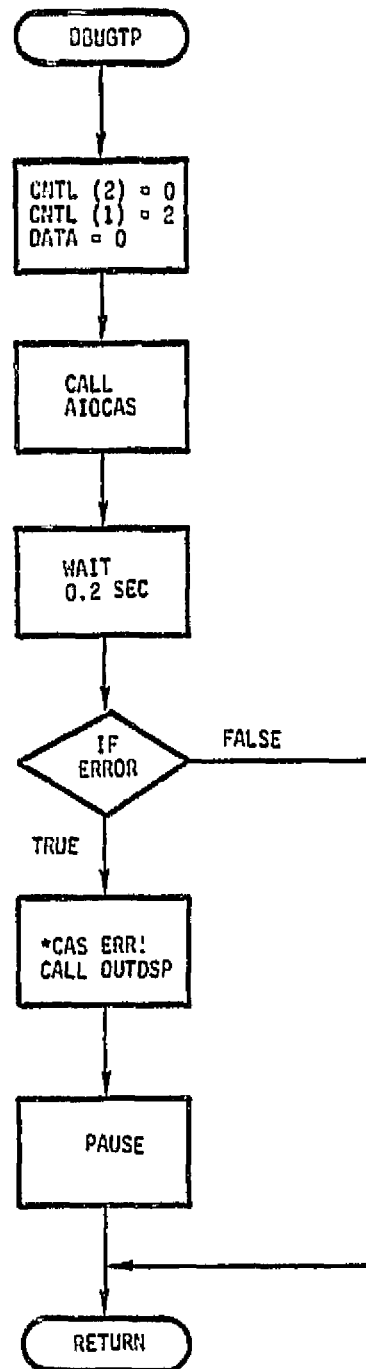
Destroys: Nothing

Modifies: TOB, DOB

Language: FORTRAN

L-V-A: Version 1, Level 1
 Roger B. Fish





SUPERVISOR EXECUTION MODULE

Subroutine: DEBUGCK

DEBUGCK checks the clock timer for VCS synchronization.

Inputs: None

Outputs: None

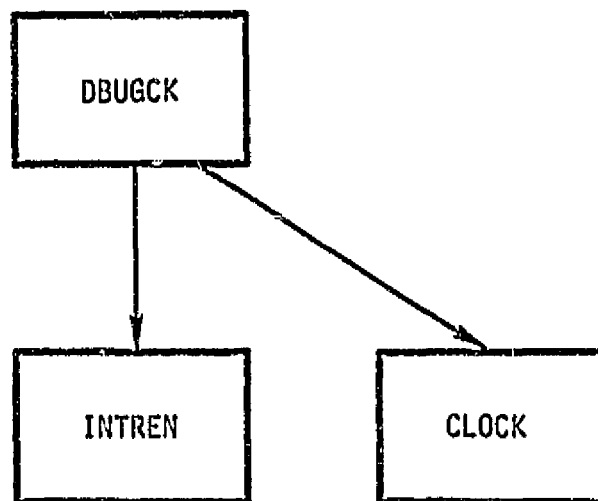
Calls: CLOCK, INTREN

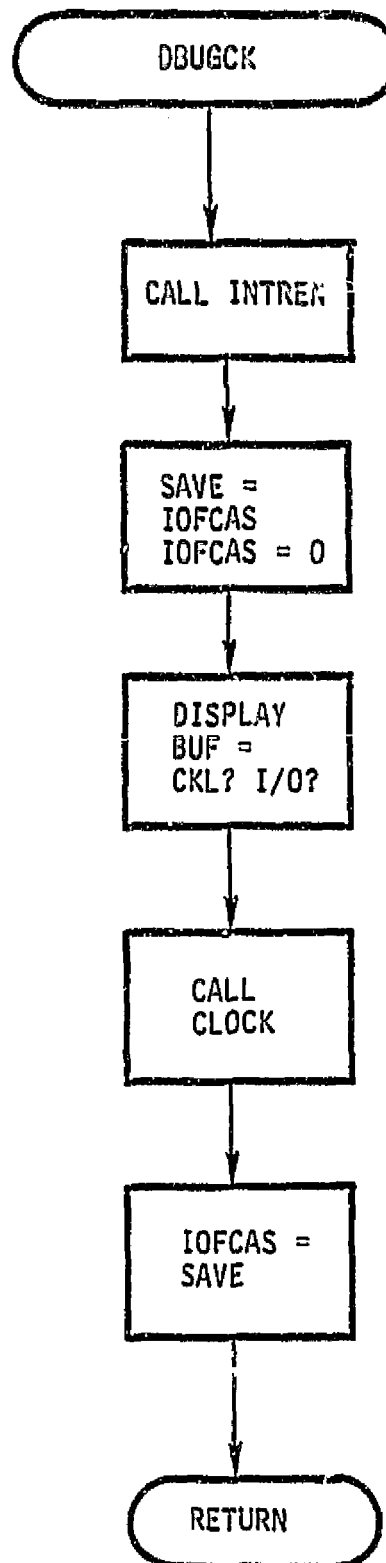
Destroys: Nothing

Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish





SUPERVISOR EXECUTION MODULE

Subroutine: DDUGME

DDUGME displays memory to the operator via LEDS.

Inputs: None

Outputs: None

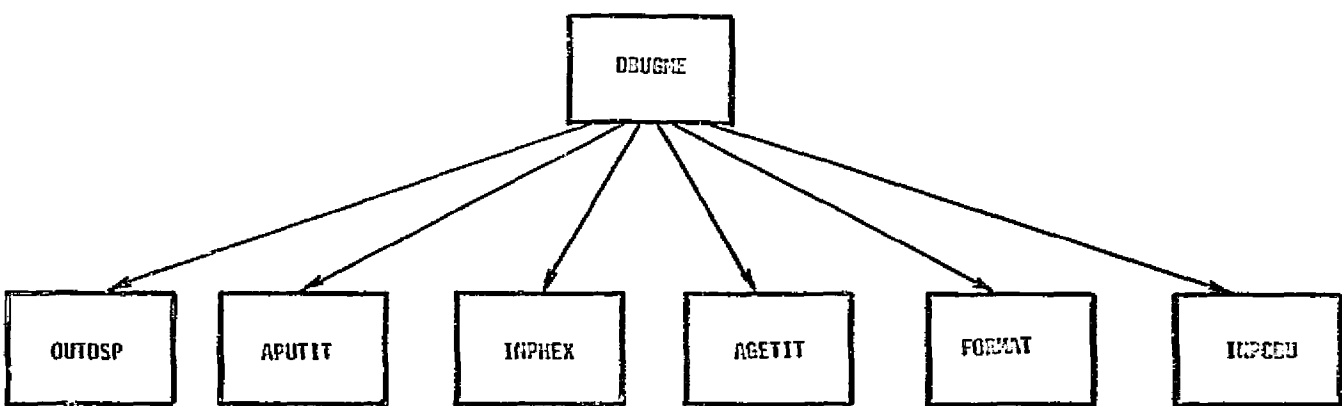
Calls: OUTDSP, INPCDU, AGETIT, INPHEX, FORMAT,
APUTIT

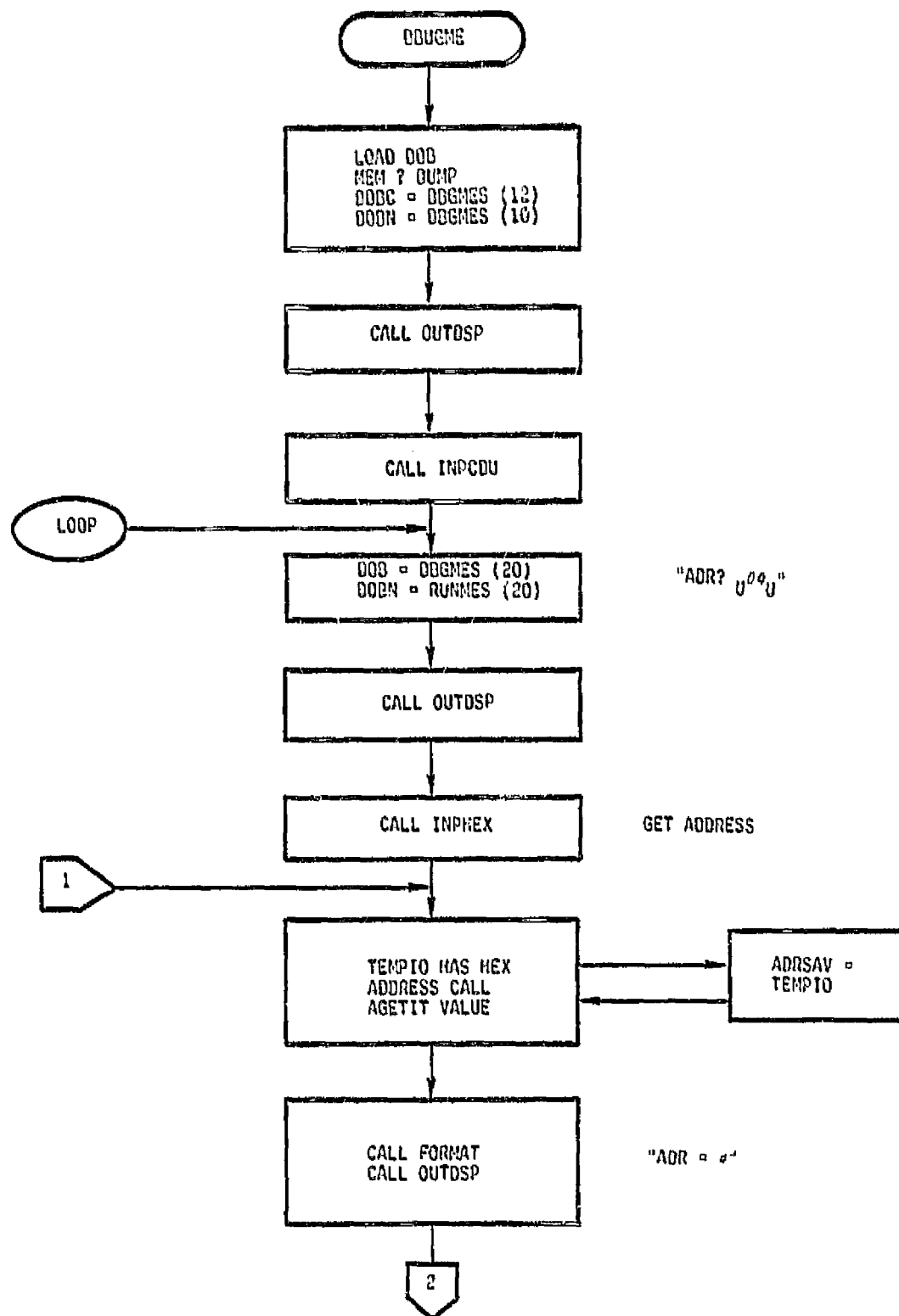
Destroys: Nothing

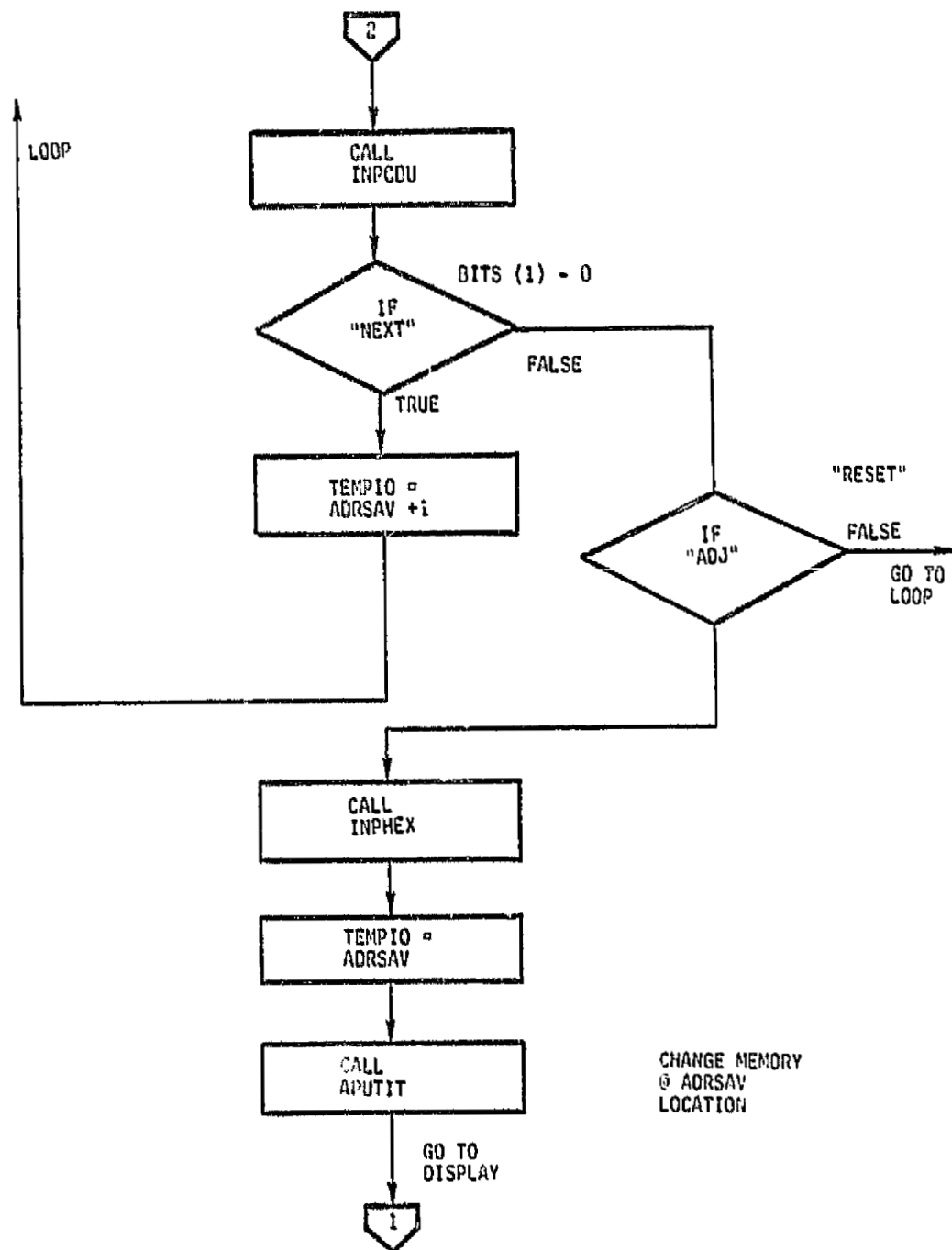
Modifies: DOB, memory location if "PUT"

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish







UTILITY ROUTINES

PAUSE	-	Wait at KEYPAD
CLOCK	-	Clock synchronizer
INPCDU	-	Read C/D panel
INPHEX	-	Input a HEX value
CONFNC	-	Convert sensor to data
RANGCK	-	Range check on sensors
SENRLM	-	Sensor rate limit test
INPCID	-	Input CID value
INPPKS	-	Input pick data
OUTCAS	-	Output data to cassette
OUTDSP	-	Display in LEDS
SAVDAT	-	Store data in CMOS and for cassette
TAPITZ	-	Initialize tape header.

I/O UTILITY ROUTINES

Subroutine: PAUSE

PAUSE waits for any key to be pressed on the display/
control.

Inputs: None

Outputs: None

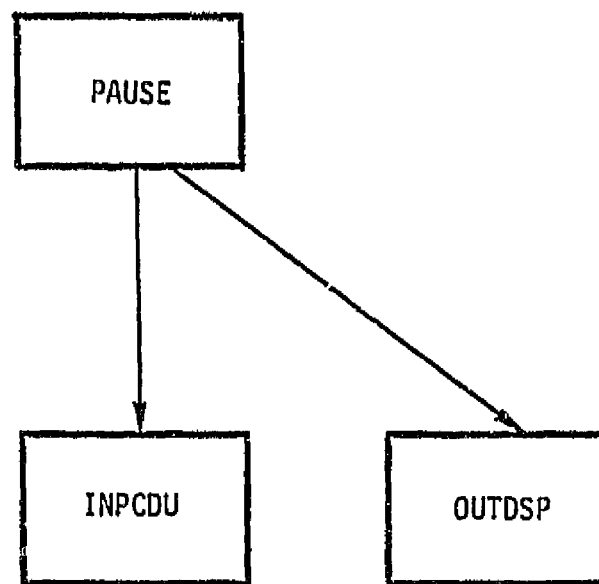
Calls: INPCDU, OUTDSP

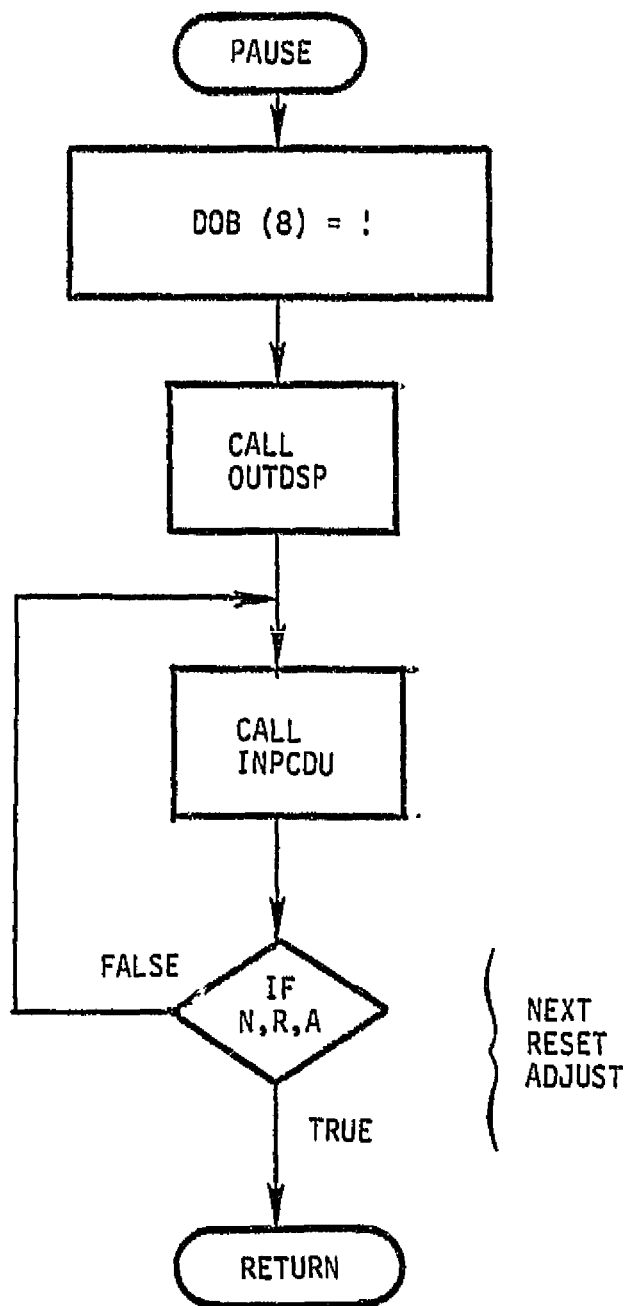
Destroys: Nothing

Modifies: DOB

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





G.P. UTILITY ROUTINES

Subroutine: CLOCK

CLOCK synchronizes VCS timing by waiting for a timer interrupt (1 second), times out after 2 seconds.

Inputs: None

Outputs: None

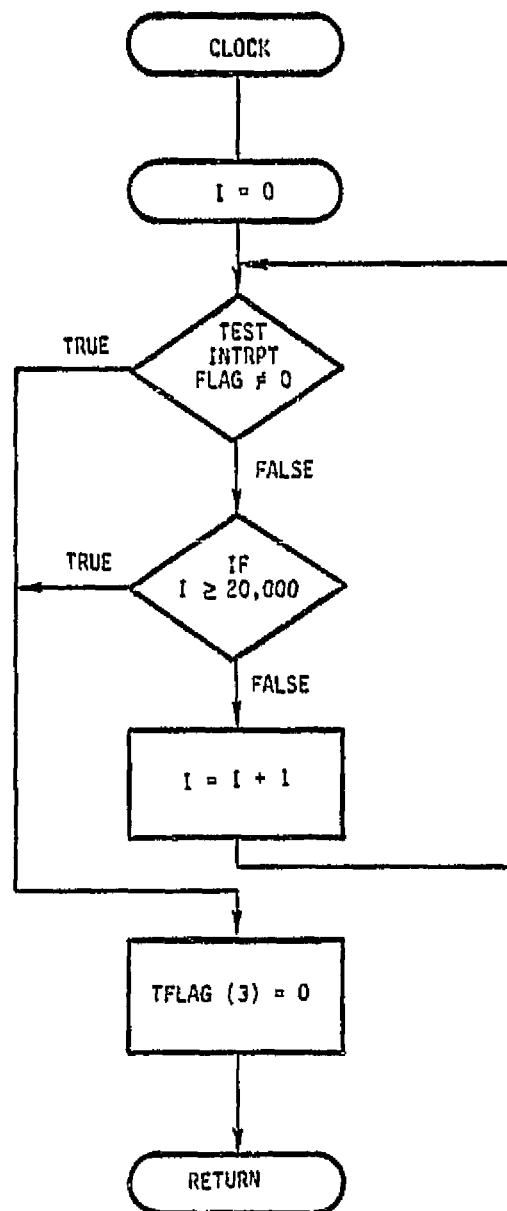
Calls: Nothing

Destroys: Nothing

Modifies: Clears TFLAG (3) = 0

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish



I/O UTILITY ROUTINES: INPUT

Subroutine: INPCDU

INPCDU inputs the data from the C/D via port 04.

Inputs: None

Outputs: C/D input in PIB

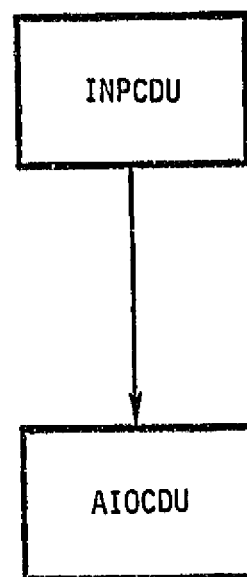
Calls: AIOCDU

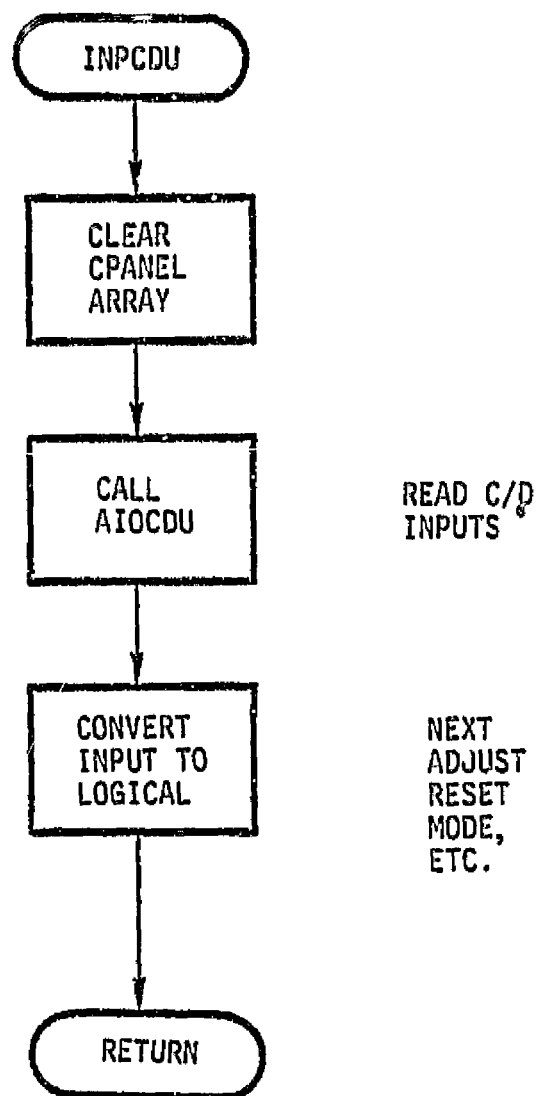
Destroys: Nothing

Modifies: /WORK/, /CPANEL/, /ERRFLG/

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





I/O UTILITY ROUTINES: INPUT

Subroutine: INPHEX

INPHEX gets a HEX number from the C/D by prompting.

Inputs: None

Outputs: HEX value in TEMPIO

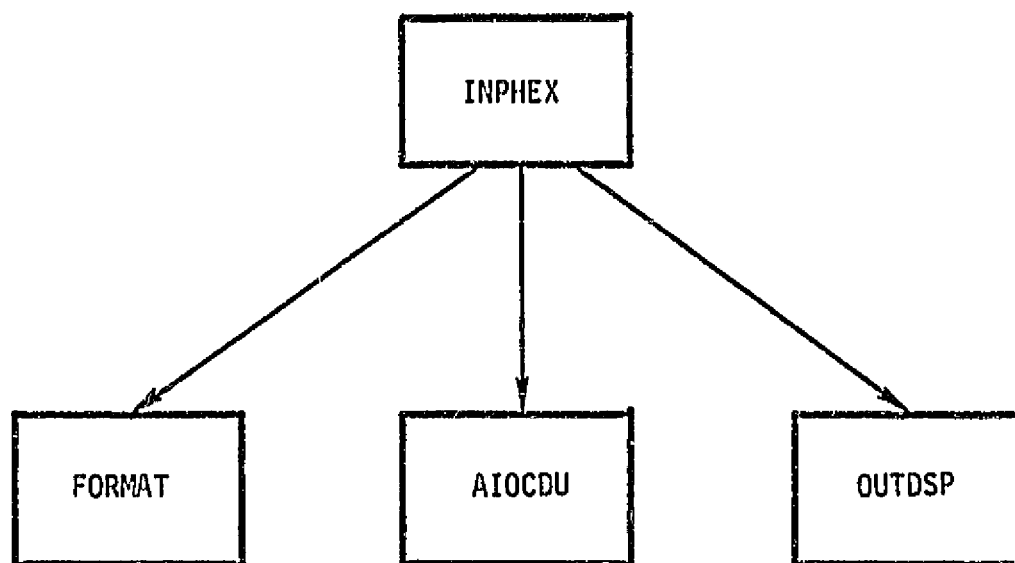
Calls: AIOCDU, OUTDSP, FORMAT

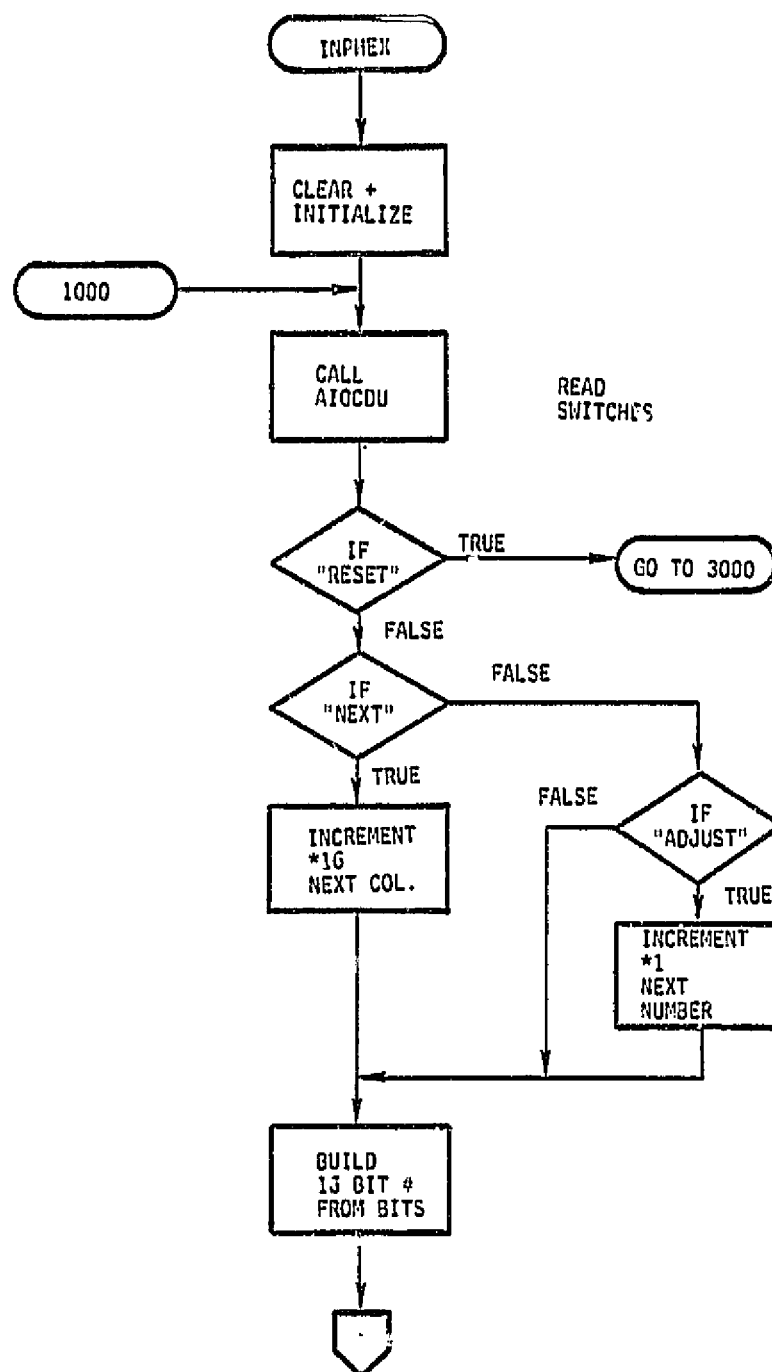
Destroys: Nothing

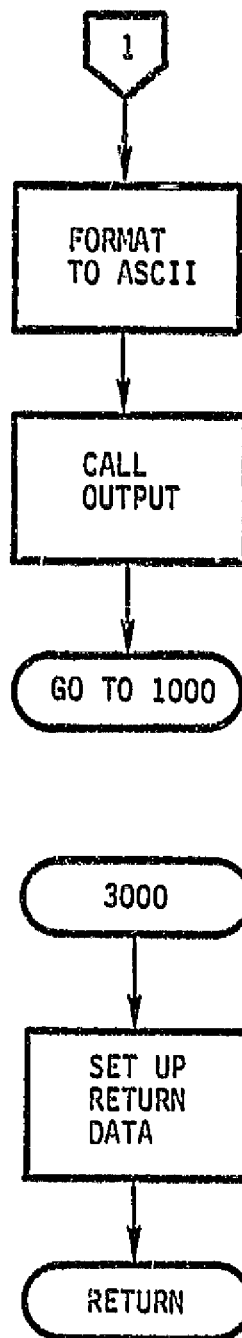
Modifies: /WORK/

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish







I/O UTILITY ROUTINES: CONVERSION

Subroutine: CONFNC

CONFNC is a table driver routine which is used to convert sensor inputs to engineering unit data.

Inputs: Sensor index SENPNT

Outputs: None

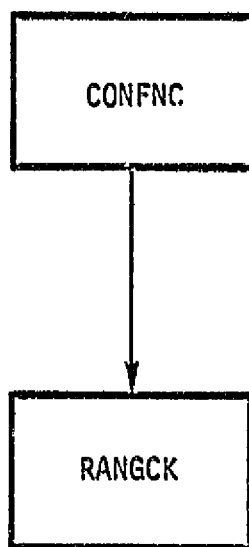
Calls: RANGCK (opt)

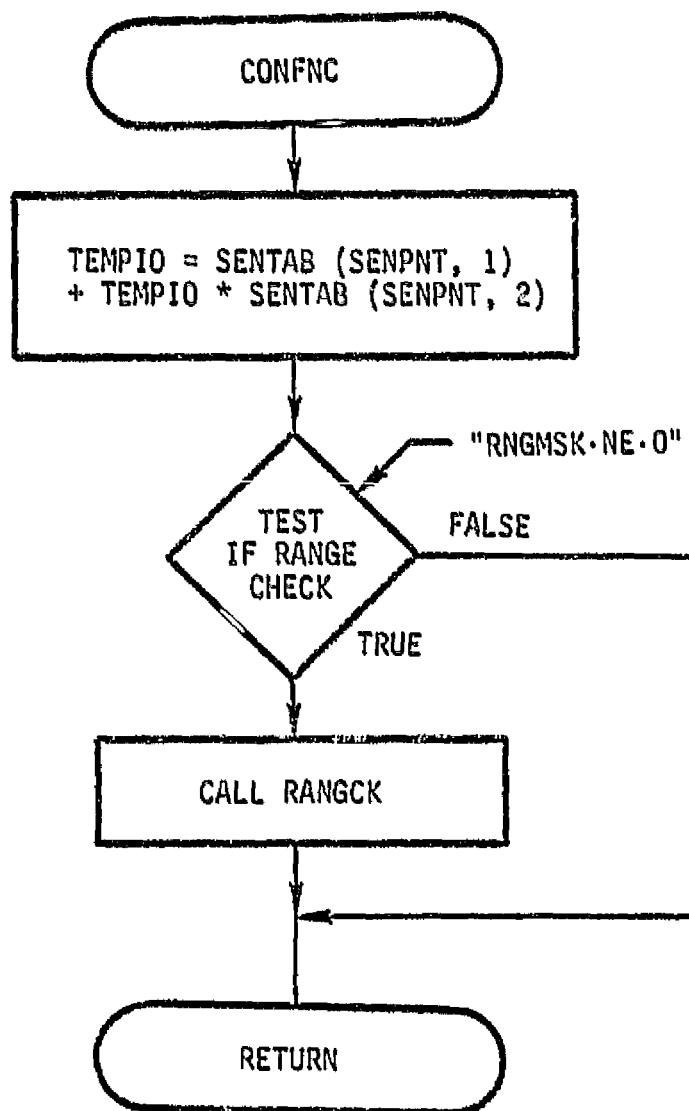
Destroys: Nothing

Modifies: TEMPIO

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





I/O UTILITY ROUTINES: CONVERSION

Subroutine: RANGCK

RANGCK test converted sensor data against max and min
stored in table

Inputs: Sensor index pointer

Outputs: ERROR FLAG 1

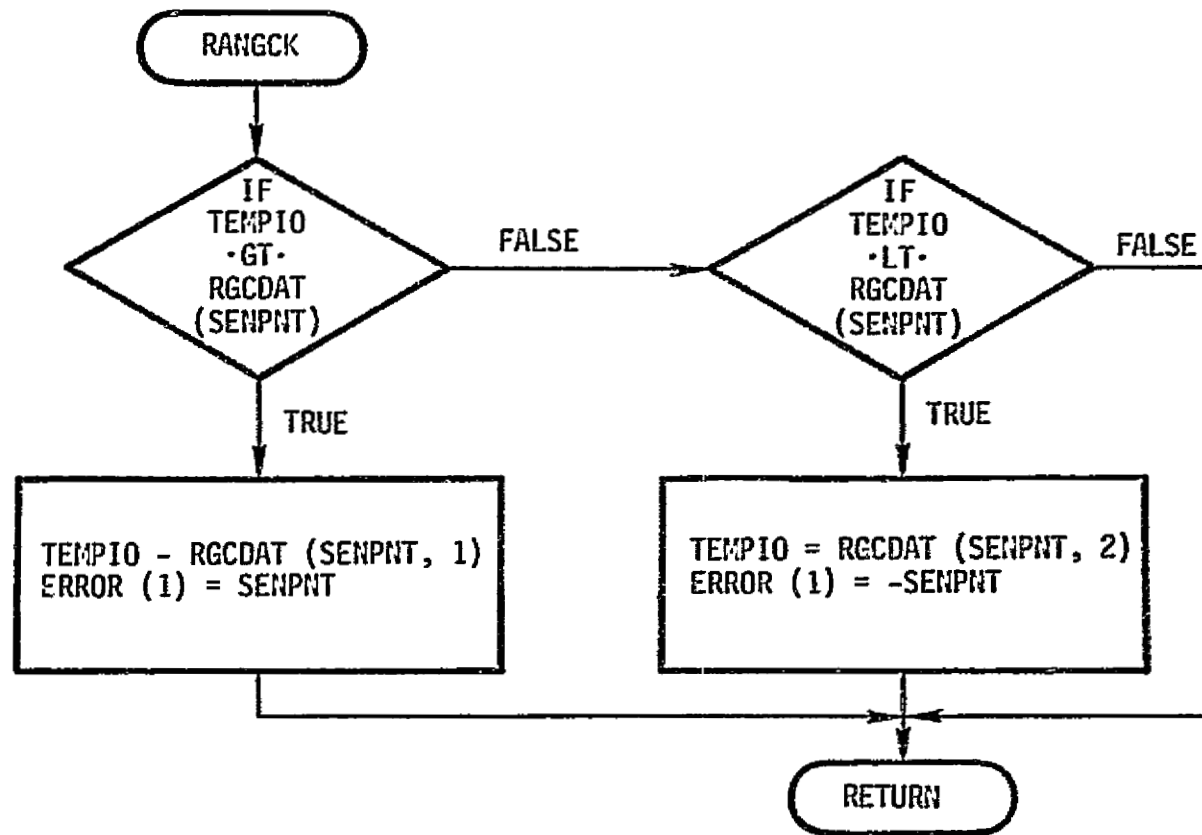
Calls: Nothing

Destroys: Nothing

Modifies: SDB, ERROR (1)

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish



I/O UTILITY ROUTINES: SENRLM

Subroutine: SENRLM

SENRLM is the reasonableness check on the LCF, PCF data.

Inputs: SDB, LCF, PCF ; STATUS

Outputs: SDB, LCF, PCF, ERROR (4)

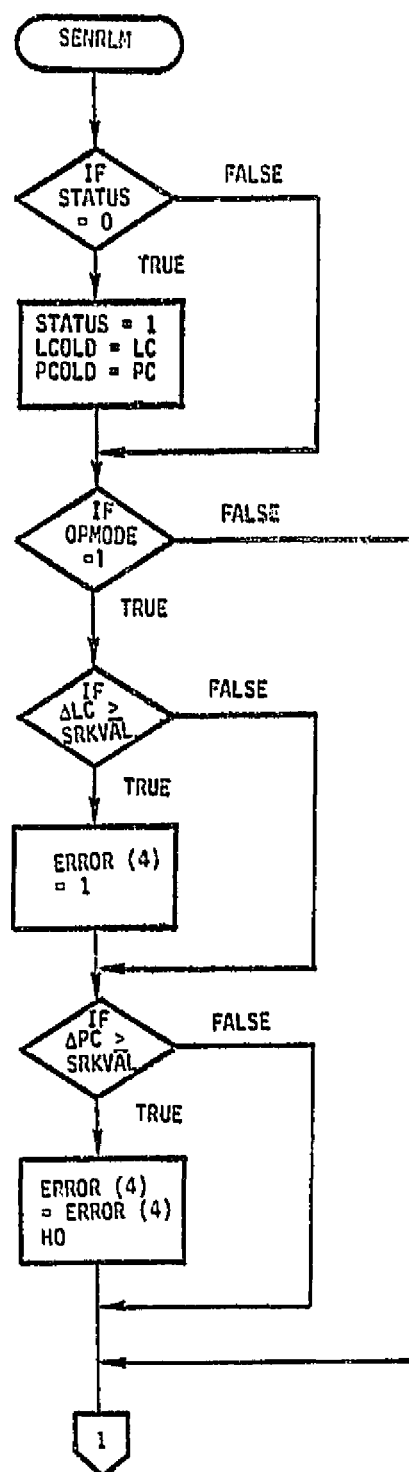
Calls: Nothing

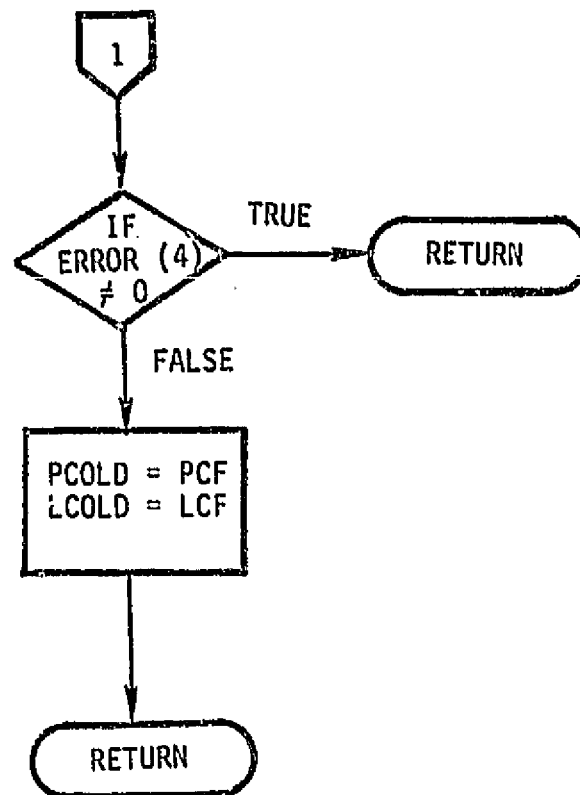
Destroys: Nothing

Modifies: LCF, PCF, ERROR (4), STATUS

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





I/O UTILITY ROUTINES: INPUT

Subroutine: INPCID

INPCID inputs the CID data via port 01, control via
port 02

Inputs: None

Outputs: CID data in SIB

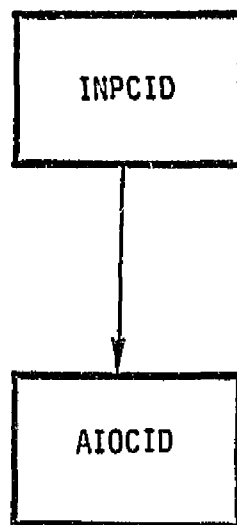
Calls: AIOCID

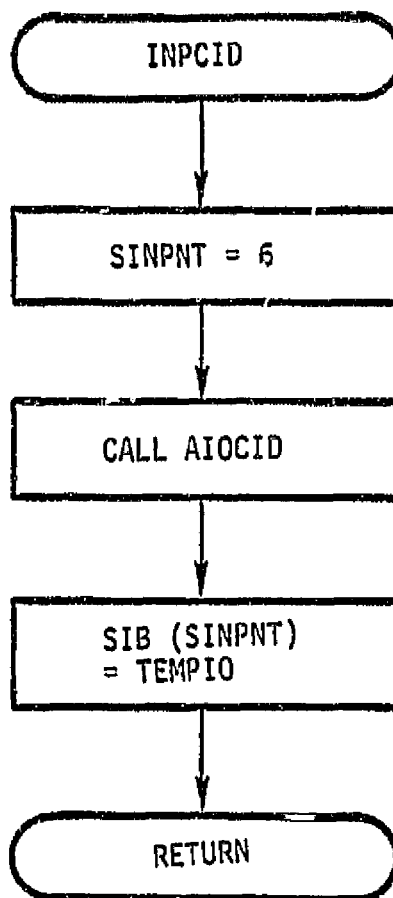
Destroys: Nothing

Modifies: SIB

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





I/O UTILITY ROUTINES: INPUT

Subroutine: INPPKS

INPPKS inputs the front pick signal via port 10H

Inputs: None

Outputs: Pick (front and rear) data in SIB

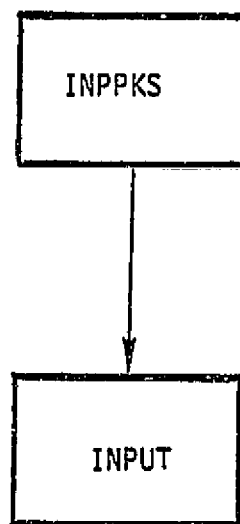
Calls: INPUT (FORTRAN BYTE I/O)

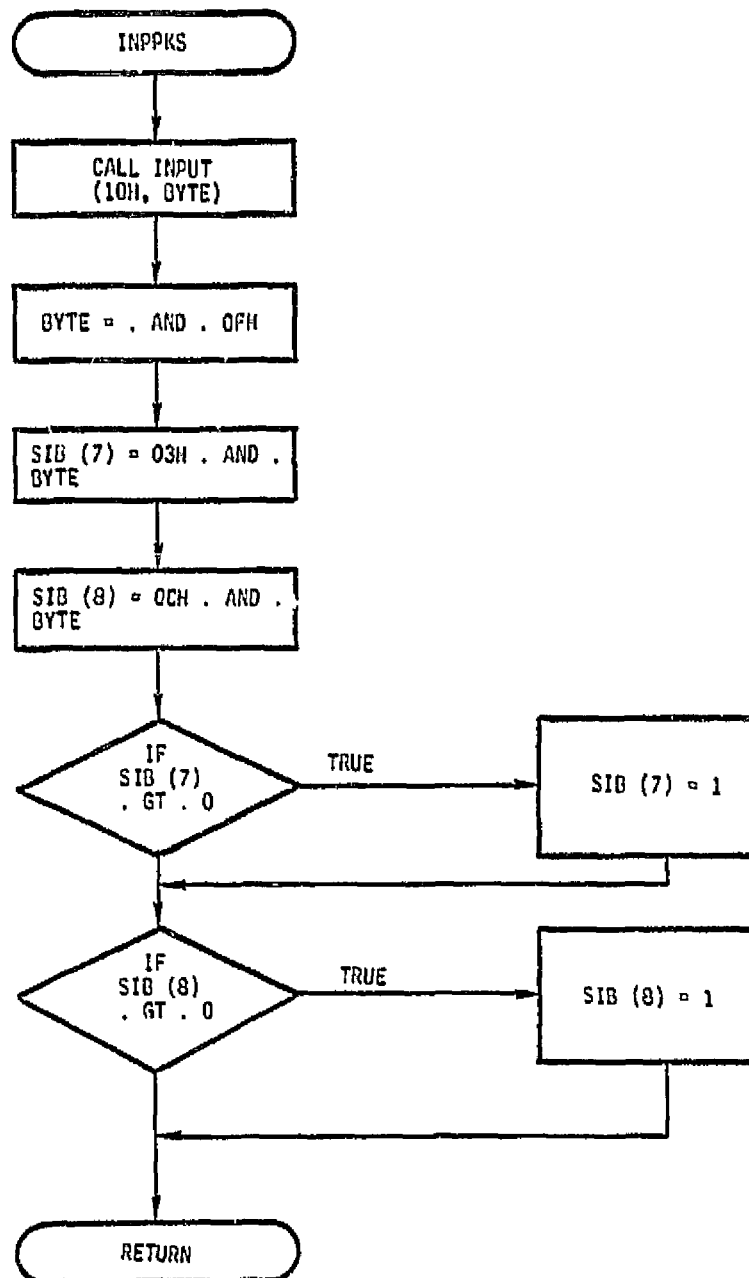
Destroys: Nothing

Modifies: SIB

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





I/O UTILITY ROUTINES: OUTPUT

Subroutine: OUTCAS

OUTCAS outputs the cassette output buffer to the digital cassette via port 08, 09, 0A, control port 0B.

Inputs: None

Outputs: TOB

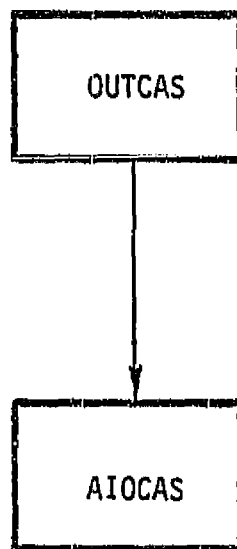
Calls: AIOCAS

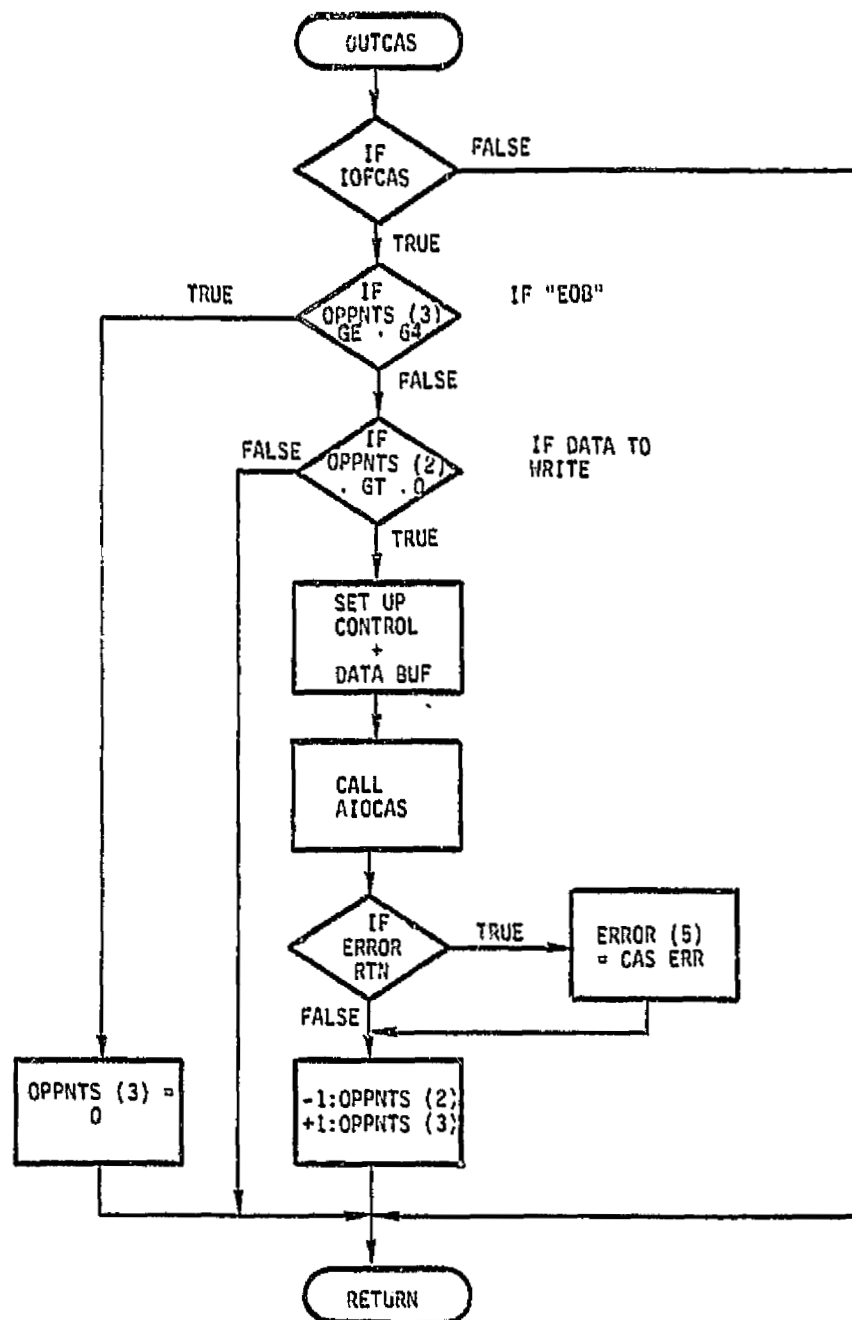
Destroys: Nothing

Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





I/O UTILITY ROUTINES: OUTPUT

Subroutine: OUTDSP

OUTDSP outputs an ASCII character string to the (7303) display.

Inputs: ASCII string in DOB

Outputs: None

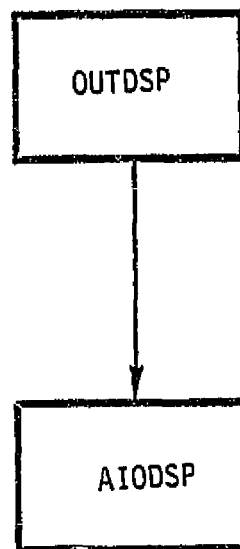
Calls: AIODSP (DOB (1))

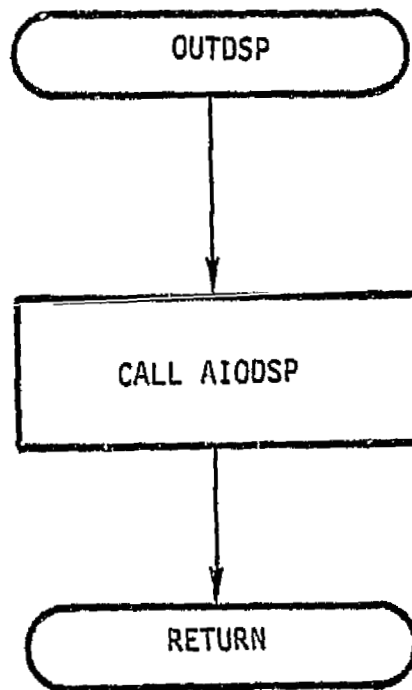
Destroys: Nothing

Modifies: Nothing

Language: FORTRAN 80

L-V-A: Version 1, Level 1
Roger B. Fish





SUPERVISOR EXECUTION MODULE

Subroutine: SAVDAT

SAVDAT stores sensor data into CASDAT buffer for tape
dump and DFD + LCF into SNDATA buffer (Mode 1).

Inputs: None

Outputs: None

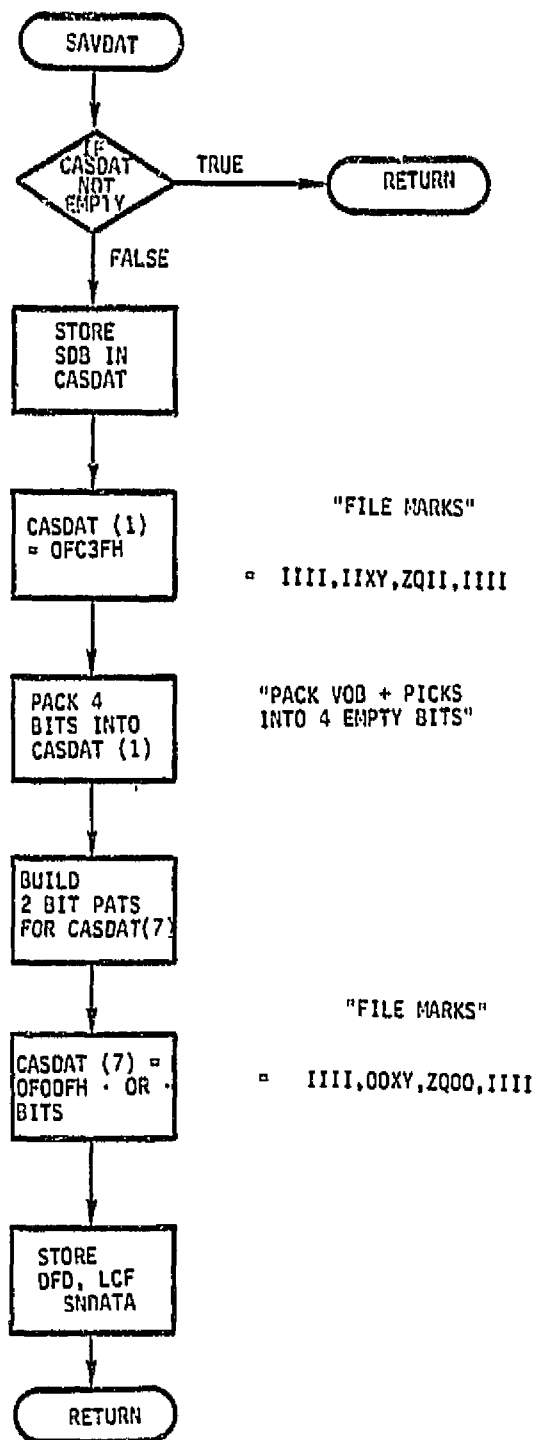
Calls: None

Destroys: Nothing

Modifies: DATPNT, SNDATA BUFFER

Language: FORTRAN 80

L-V-A: Version 1, Level 1
 Roger B. Fish



I/O UTILITY ROUTINE: OUTPUT

Subroutine: TAPITZ

TAPITZ writes the data tape file header.

Inputs: None

Outputs: None

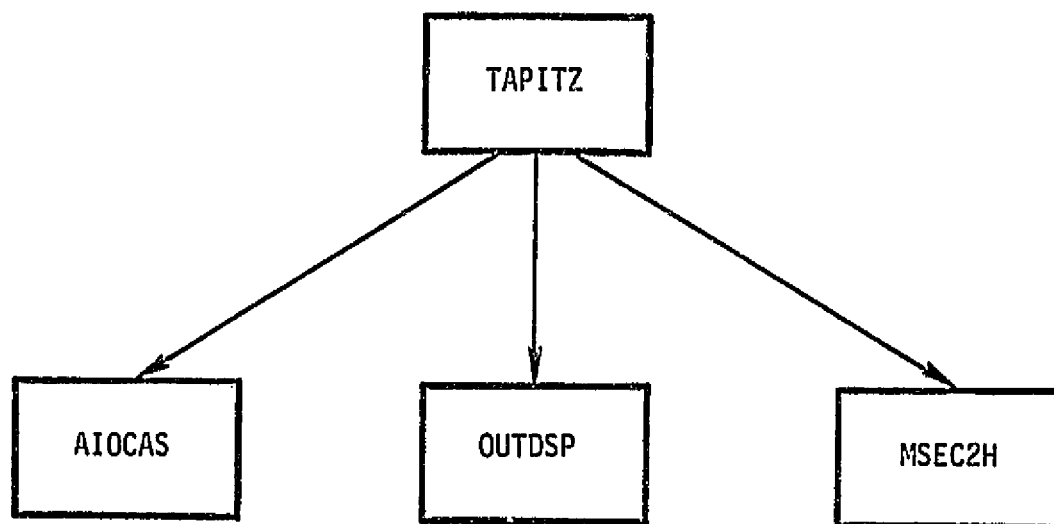
Calls: AIOCAS, OUTDSP, MSEC2H

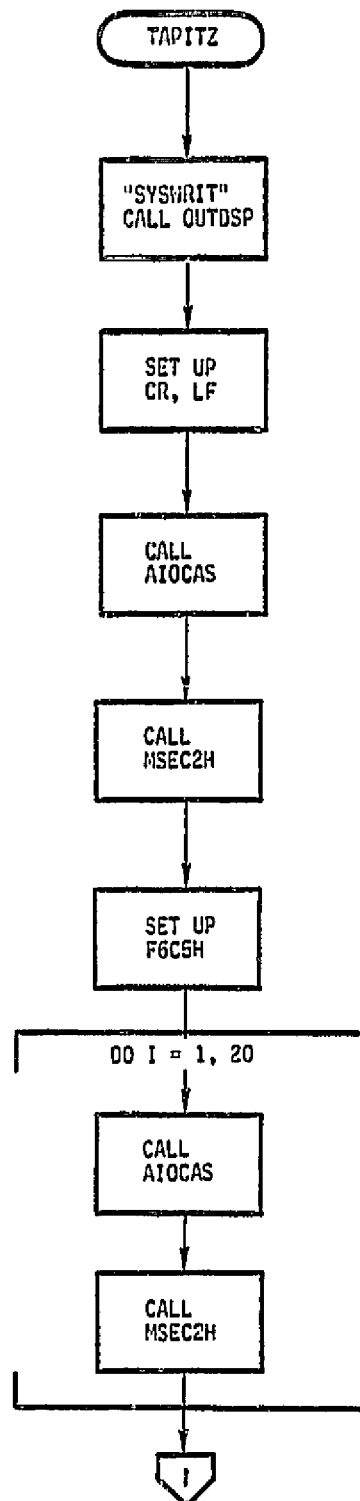
Destroys: WORK/

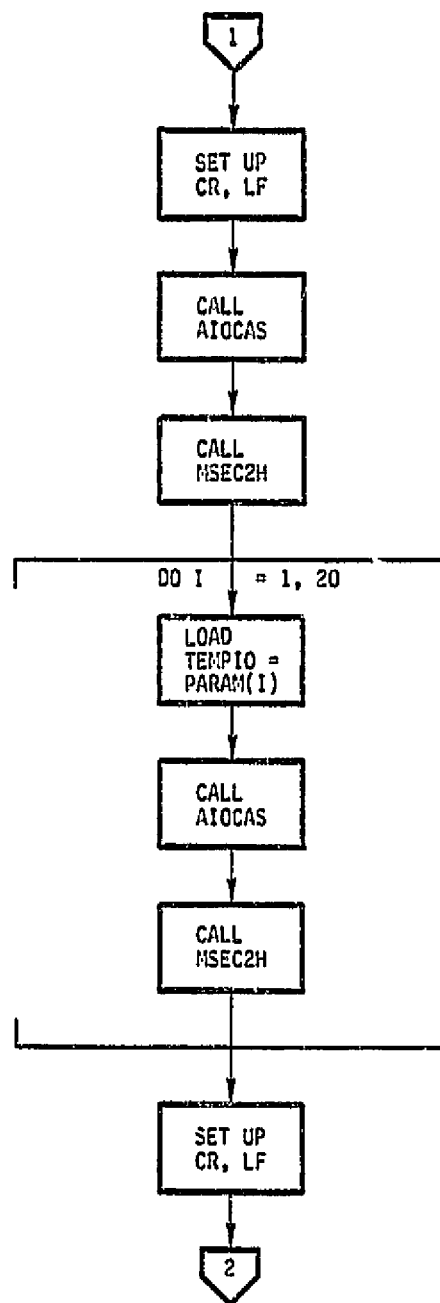
Modifies: DOB

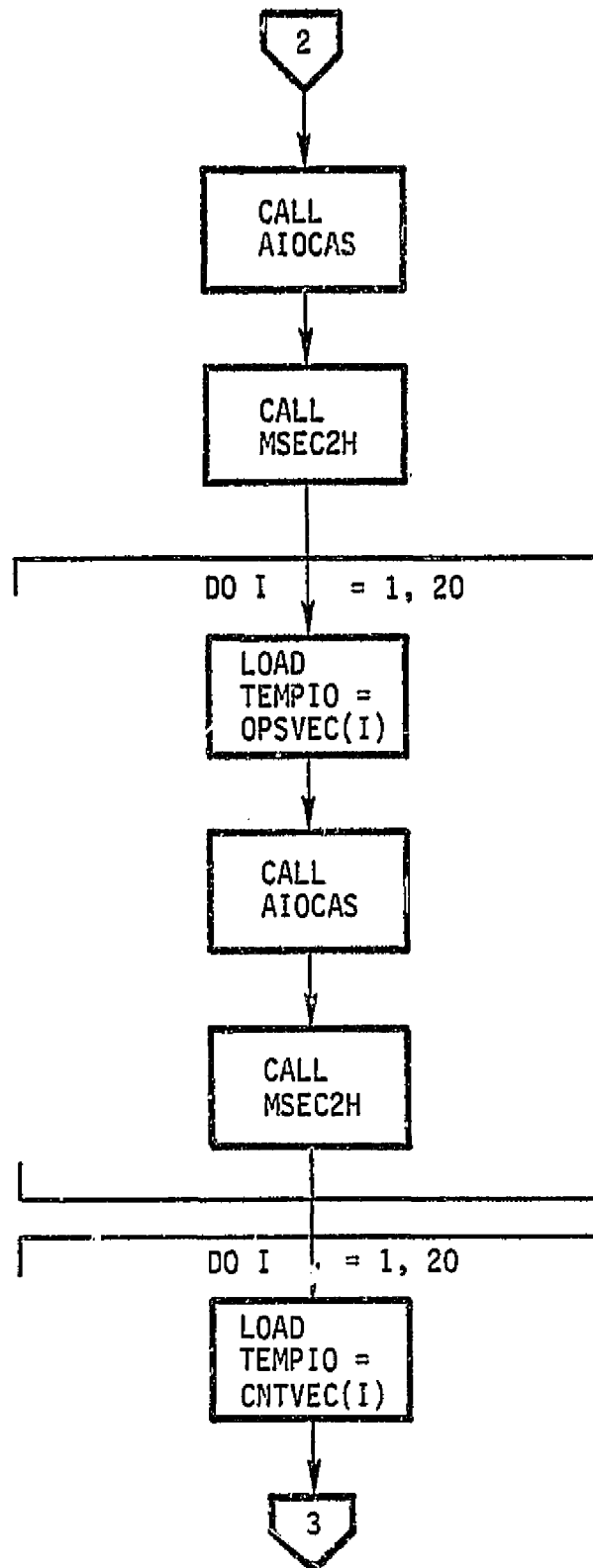
Language: FORTRAN 80

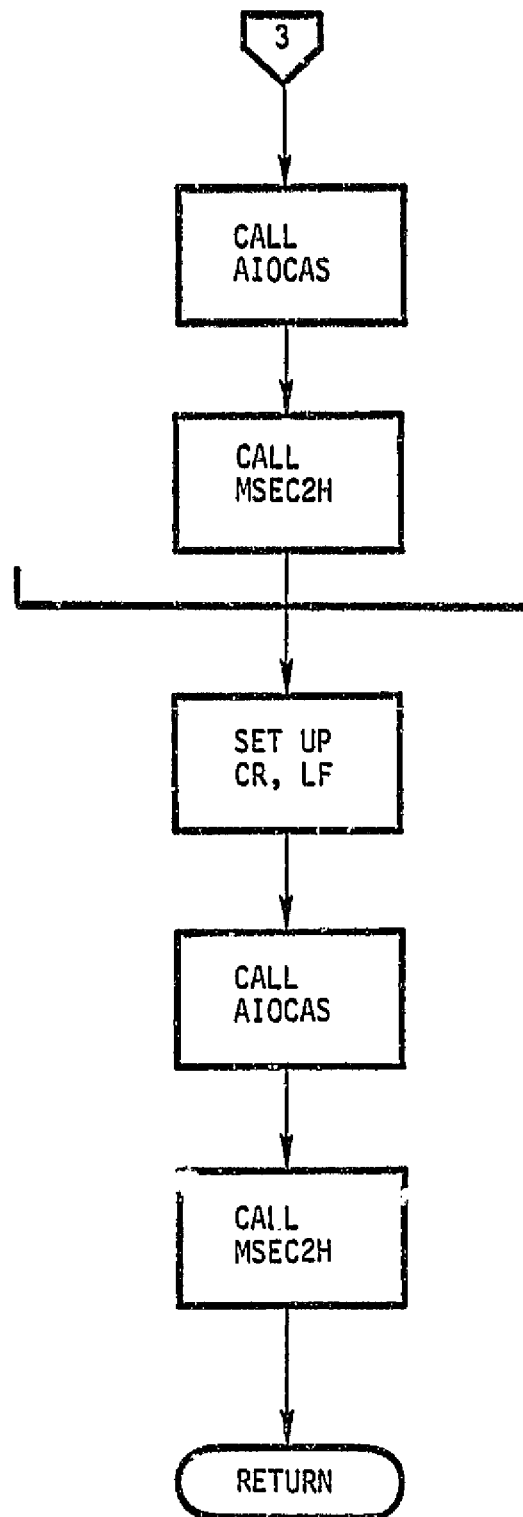
L-V-A Version 1, Level 1
 Roger B. Fish











I/O DRIVERS + INTERRUPT ROUTINES

AIODSP	-	Display call to monitor
FORMAT	-	Format # into HEX
APUTIT	-	Put a BYTE into memory
AIOCID	-	CID I/O driver
AIOCAS	-	Cassette I/O driver
AIOVOT	-	Valve I/O driver
AIOITZ	-	I/O initializer routine
AIOADC	-	A/D I/O driver
AGETIT	-	Get a BYTE from memory
AIOCDO	-	C/D I/O driver
INTRPT	-	Interrupt handler (timer)
INTREN	-	Interrupt enable
INTRDS	-	Interrupt disable

I/O DRIVER ROUTINES:

Subroutine: AIODSP (DOB (1))

AIODSP writes ASCII data to the display unit (7303)
via the monitor from DOB

Inputs: Address of DOB (1) in BC

Outputs: None

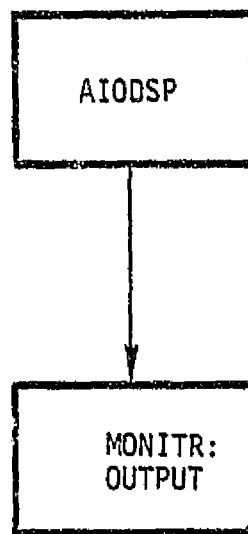
Calls: MONITOR : OUTPT

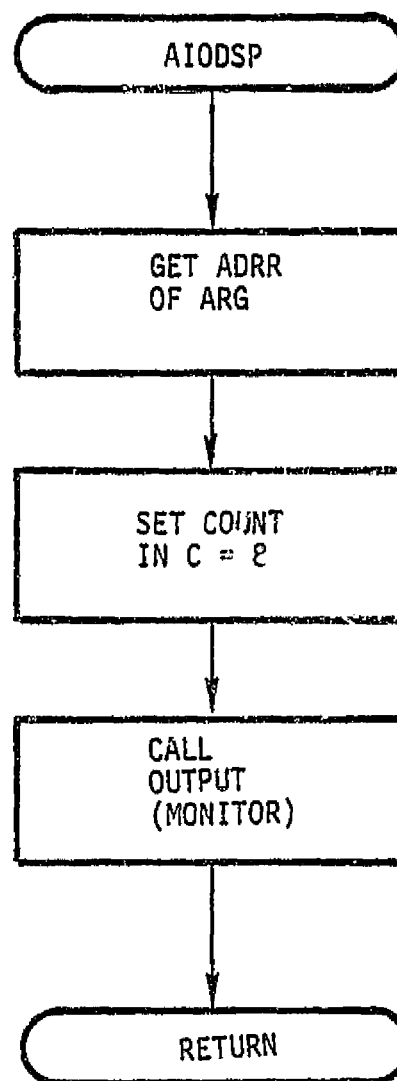
Destroys: F/F's, A

Modifies: Nothing

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level
Roger B. Fish





I/O UTILITY ROUTINES:

Subroutine: FORMAT (ADDR)

FORMAT turns a HEX value in DOB (5,6) to ASCII code,
in (BC 5, 6, 7, 8)

Inputs: None

Outputs: None

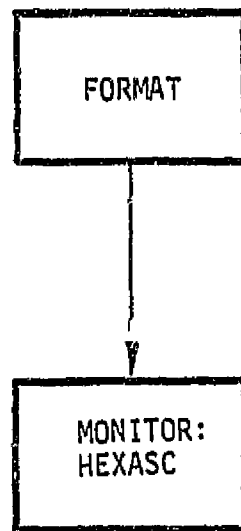
Calls: MONITOR: HEXASC

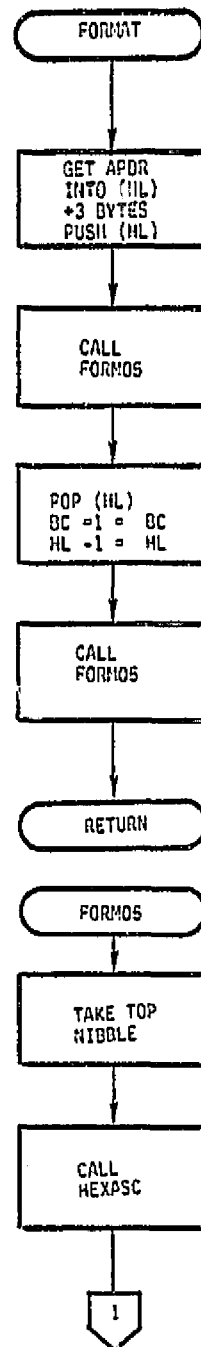
Destroys: A, F/F, BC, HL

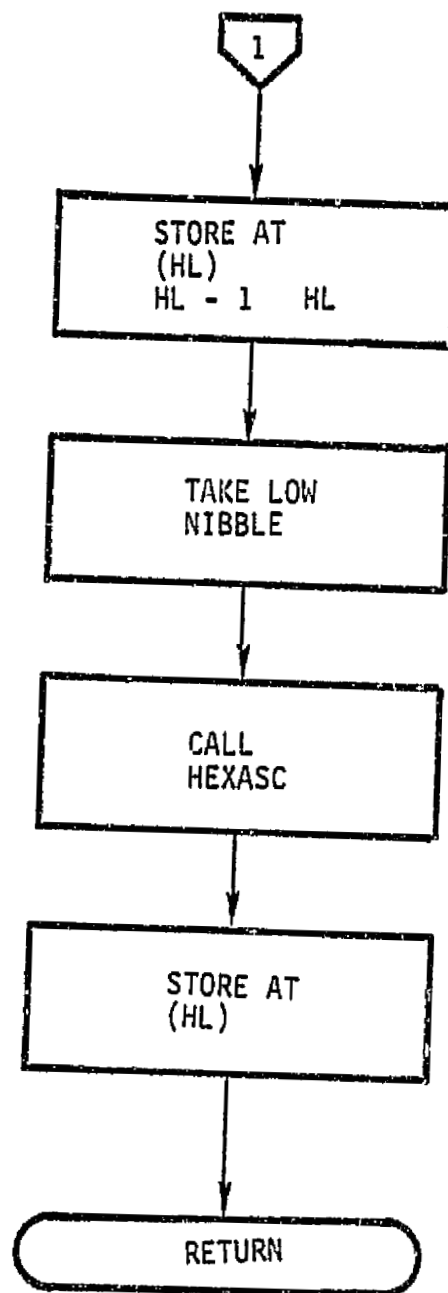
Modifies: DOB

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
Roger B. Fish







I/O UTILITY ROUTINES:

Subroutine: APUTIT (TEMPIO)

APUTIT stores a BYTE in memory complementary to AGETIT.

Inputs: Address in TEMPIO and BYTE @ TEMPIO + 1

Outputs: None

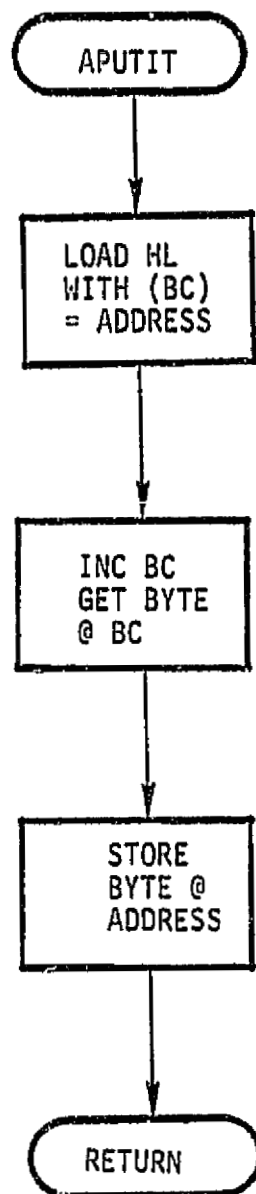
Calls: Nothing

Destroys: A, BC, F/F's, HL

Modifies: BYTE @ TEMPIO

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
 Roger B. Fish



I/O DRIVER ROUTINES:

Subroutine: AIOCID (I)

AIOCID reads the CID data from port 01, into TEMP10

Inputs: None

Outputs: None

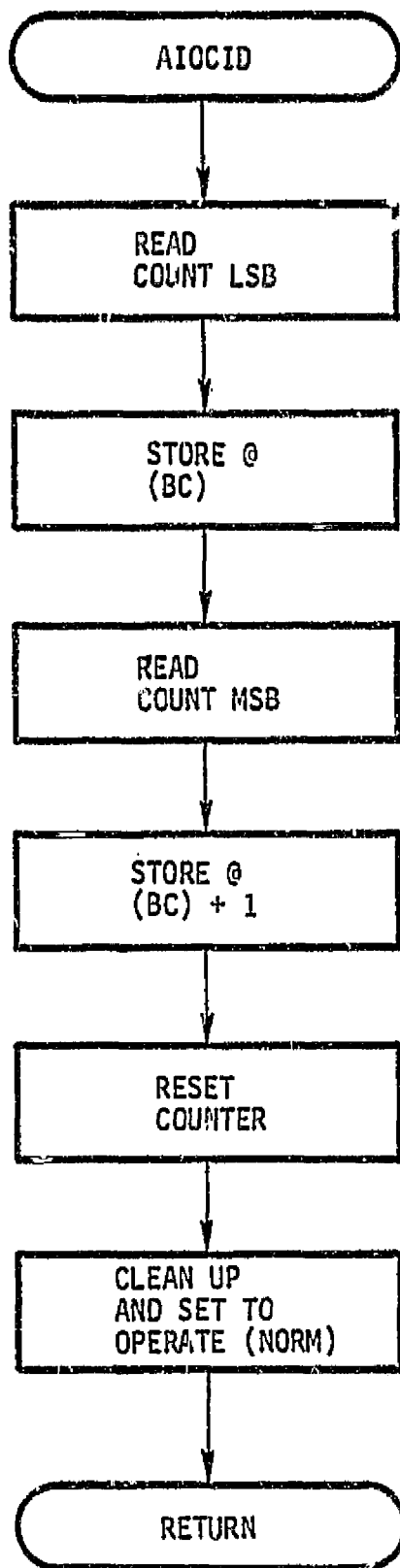
Calls: Nothing

Destroys: F/F's, A

Modifies: TEMP10

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
 Roger B. Fish



I/O DRIVER ROUTINES:

Subroutine: AIOCAS (BUF)

AIOCAS writes data from the cassette output buffer to
tape.

Inputs: None

Outputs: None

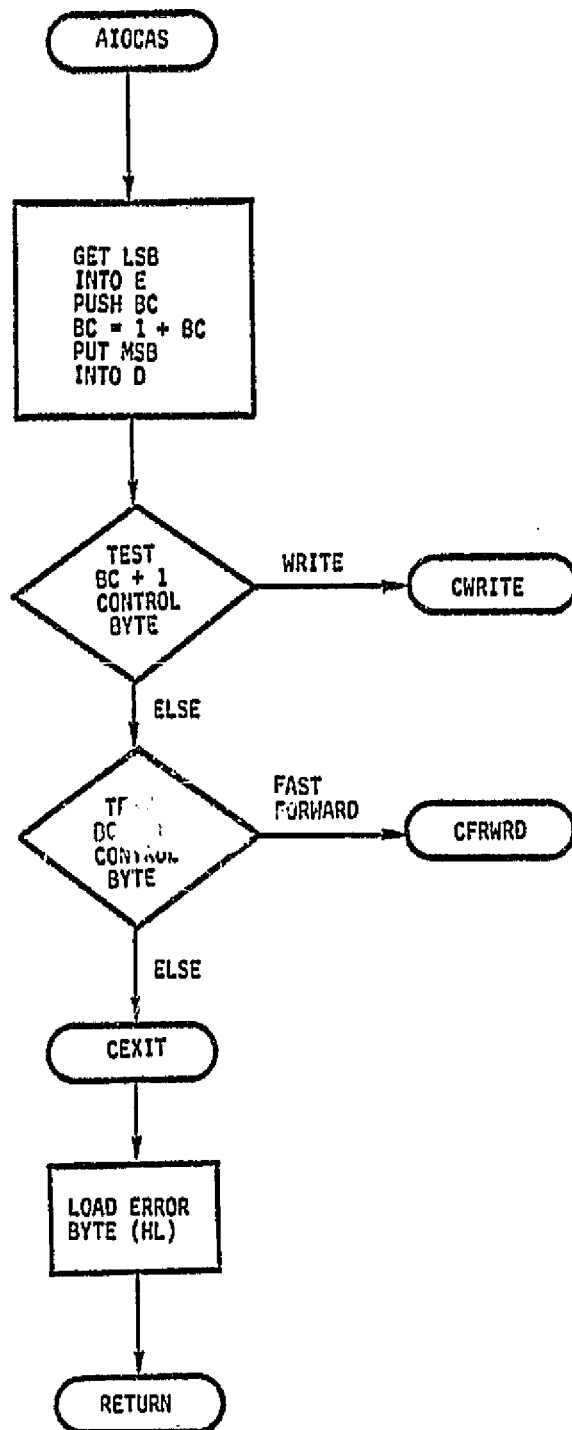
Calls: Nothing

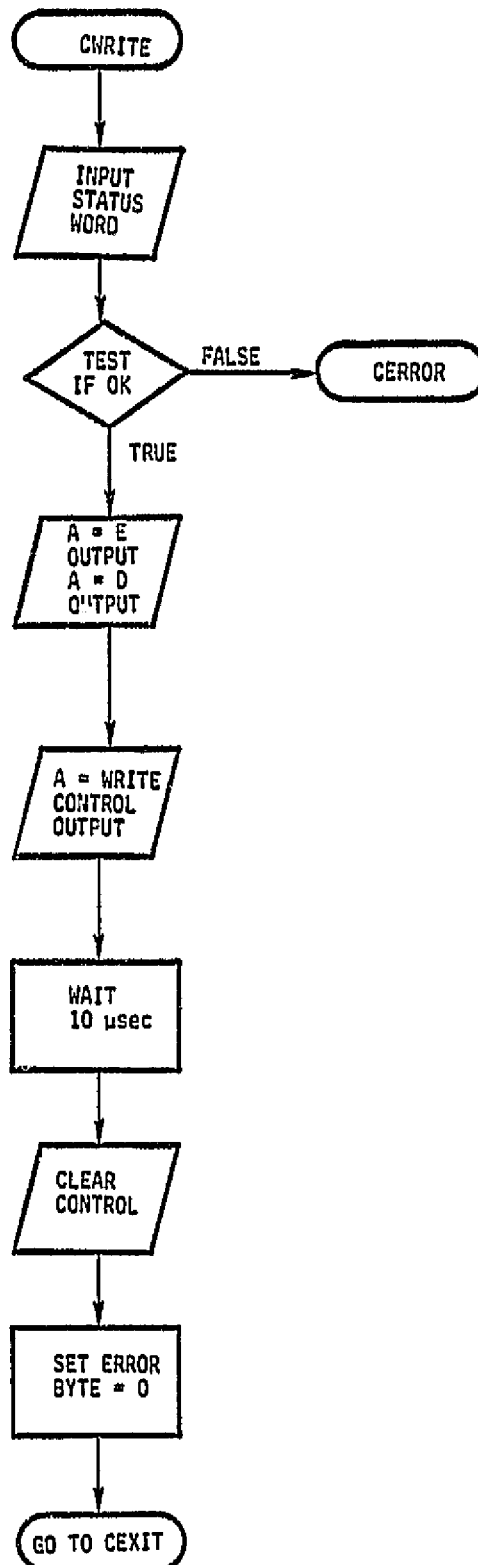
Destroys: F/F's, A

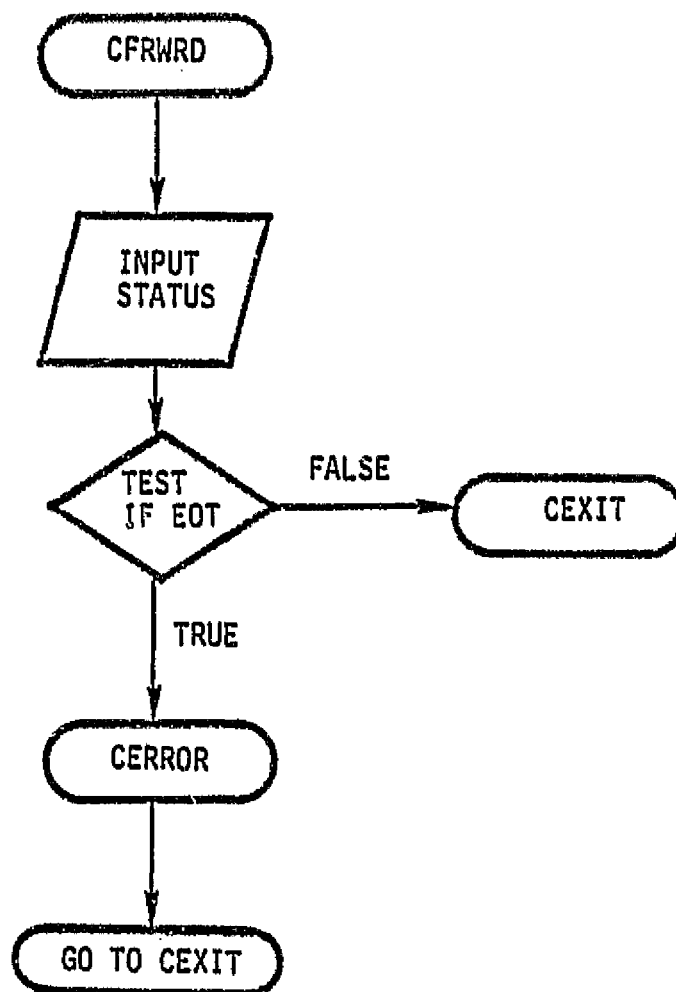
Modifies: Nothing

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
 Roger B. Fish







I/O DRIVER ROUTINES:

Subroutine: AIOVOT

AIOVOT is the valve control I/O driver routine.

Inputs: CBYTE control word

Outputs: Valve control interface lines

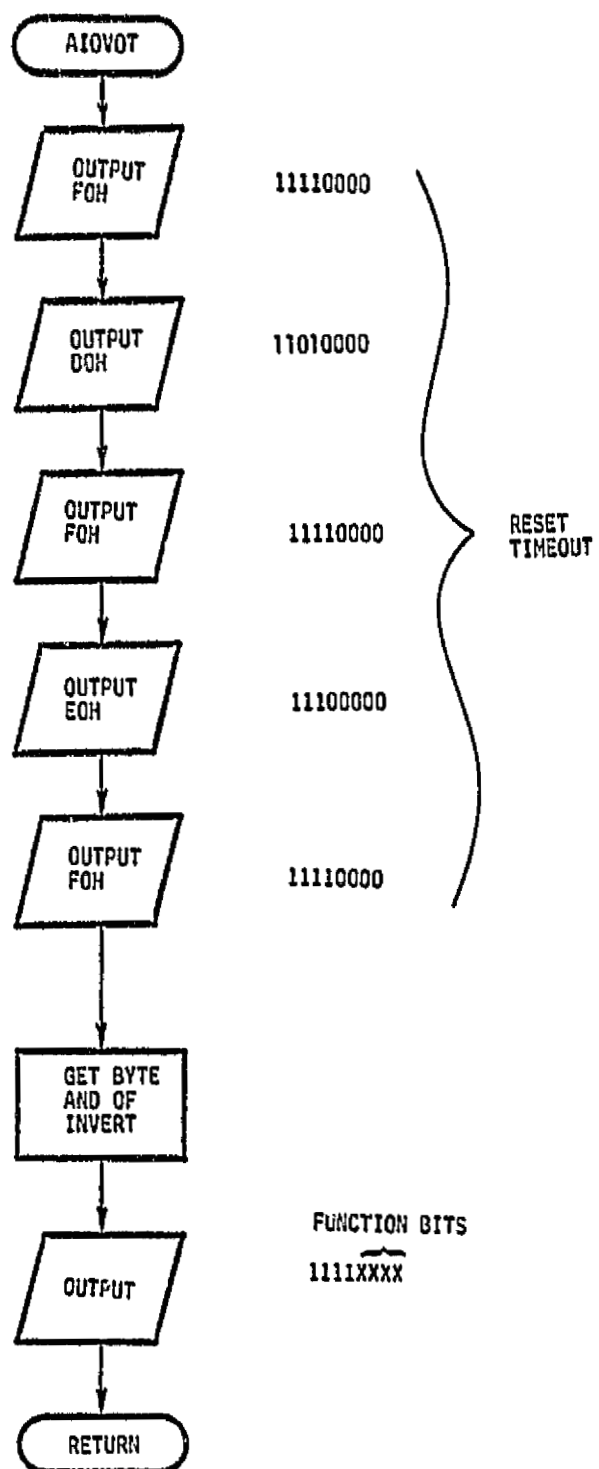
Calls: Nothing

Destroys: A, F/F's, BC

Modifies: Valve control lines

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
Roger B. Fish



I/O DRIVER ROUTINES:

Subroutine: AIOITZ

AIOITZ initializes I/O interfaces for operation at
machine startup

Inputs: None

Outputs: None

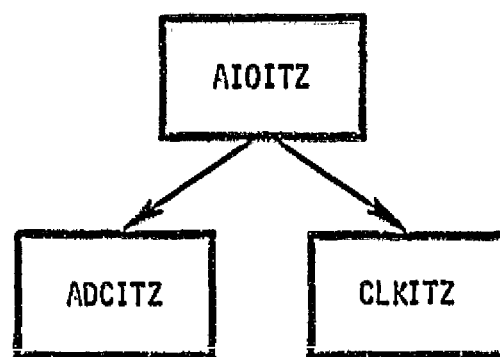
Calls: ADCITZ, CLKITZ (local subroutines)

Destroys: A, F 'F's, BC, DE, HL

Modifies: Interface status

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
 Roger B. Fish



I/O DRIVER ROUTINES:

Subroutine: AIOADC (I, J)

AIOADC reads signals from the A/D converter into the
SIB

Inputs: Sensor index, output data

Outputs: Sensor data to TEMP10

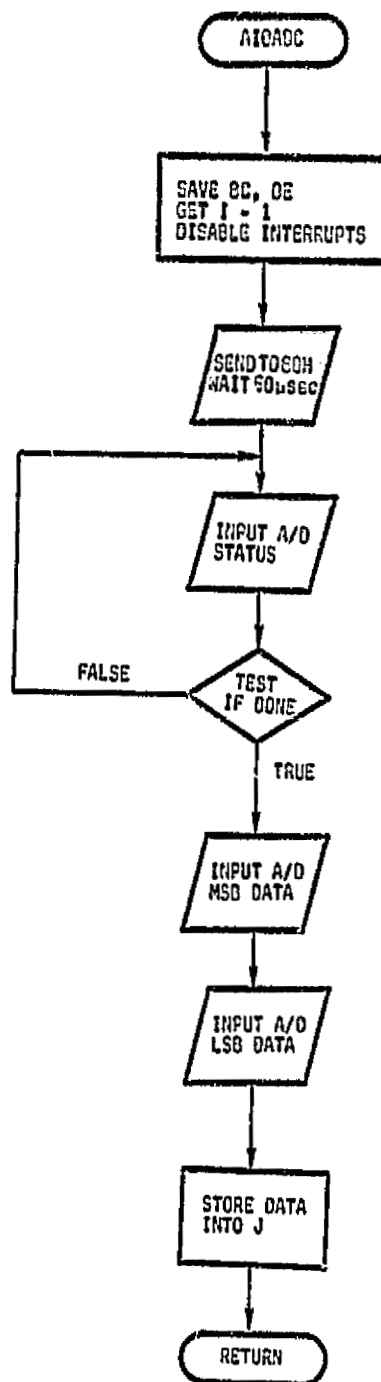
Calls: Nothing

Destroys: F/F's

Modifies: TEMP10 = /WORK/

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
Roger B. Fish



G.P. UTILITY ROUTINES:

Subroutine: AGETIT (I, J)

AGETIT takes the HEX address of TEMP10 in BC loads
the value and puts it in address held in DE.

Inputs: I = HEX

Outputs: J = HEX value

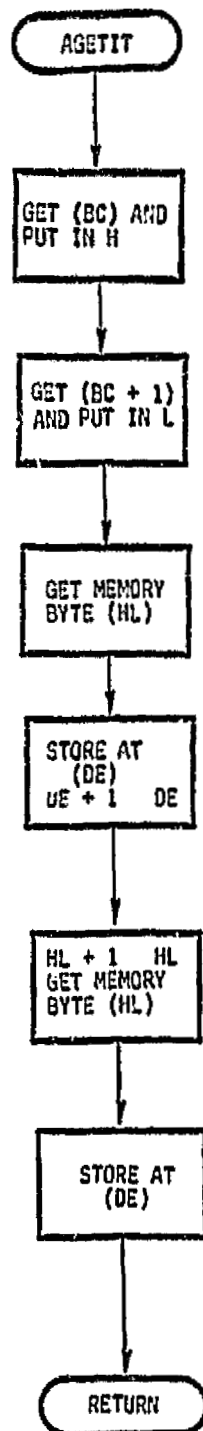
Calls: Nothing

Destroys: BC, DE

Modifies: DOB

Language: ASSEMBLY 8080/8085

L-V-A: Version 1, Level 1
Roger B. Fish



I/O DRIVER ROUTINES:

Subroutine: AIOCDU (BUF)

AIOCDU reads data from the C/D panel port 04.

Inputs: (BC) points to work buffer

Outputs: None

Calls: None

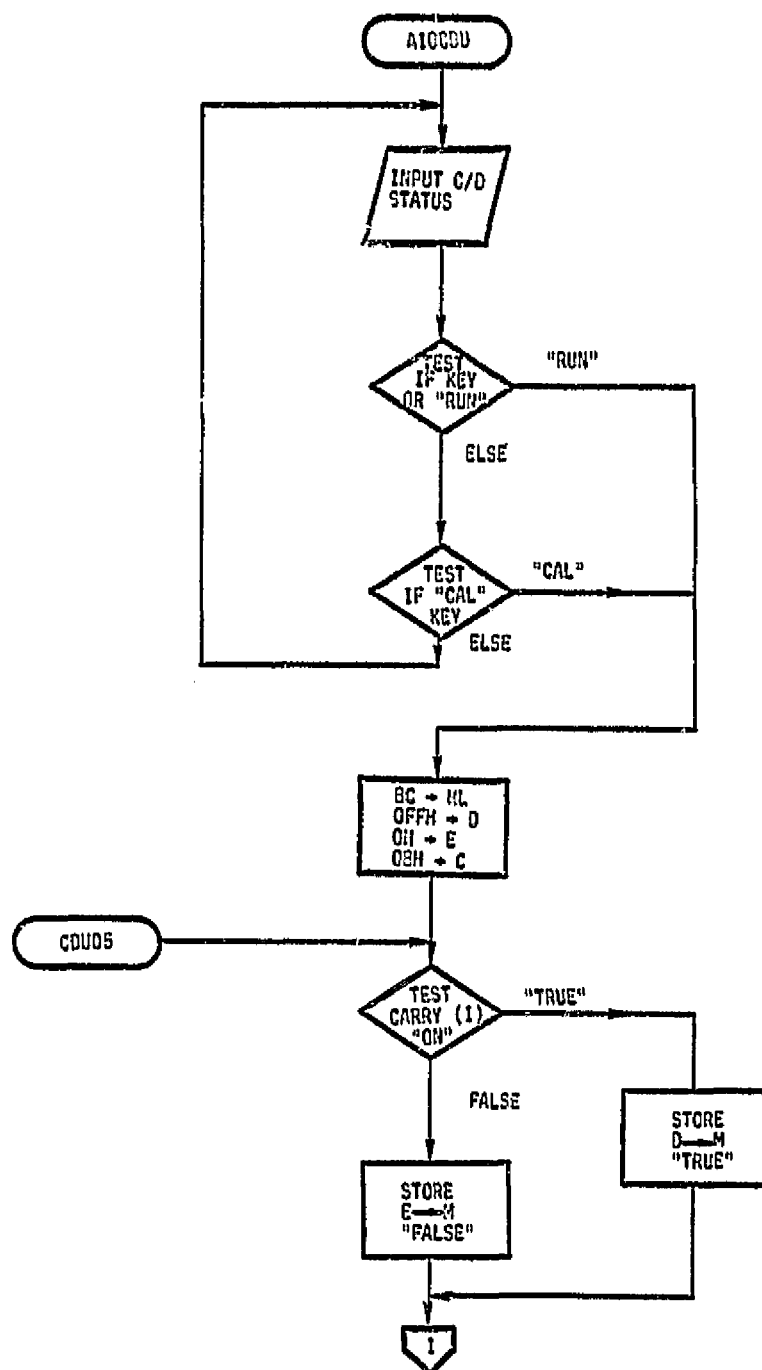
Destroys: A, F/F's, DE, HL

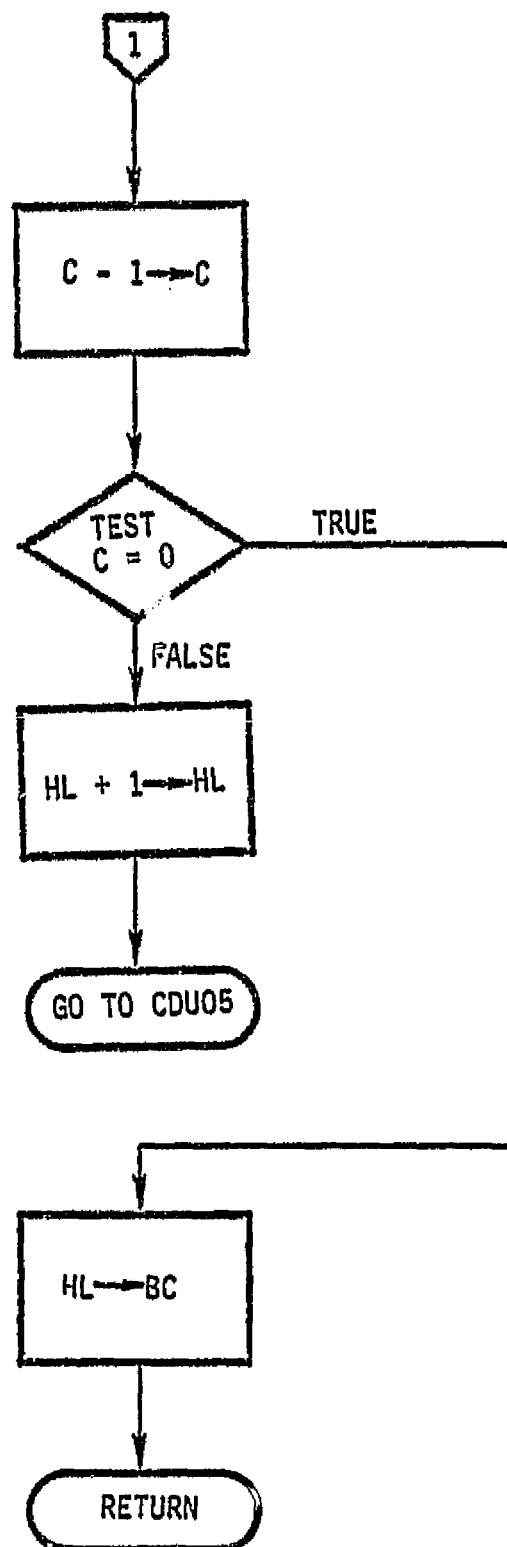
Modifies: Work vector

Description: Generator 8 bytes of local data BUF

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
Roger B. Fish





I/O UTILITY ROUTINES: INTERRUPT

Subroutine: INTRPT

INTRPT is the clock interrupt service routine.

Inputs: None

Outputs: None

Calls: OUTCAS

Destroys: Nothing

Modifies: Interrupt counters and flags in monitor code

Language: ASSEMBLER 8080/8085

I-V-A: Version 1, Level 1
 Roger B. Fish

CASSETTE INTERRUPT STRUCTURE

```
Monitor:      RST ~ 5.5
               Call INTRPT
               RET
               :
               :

INTRPT:       Read clock port
               Test if 1 second
               Test if 100 msec
               Test if 1 msec

ONESEC:       Set flag BYTE (sec)
               Add 1 to count (sec)
               RET

HUNDRED:      Set flag BYTE (100 msec)
               Call OUTCAS
               RET

MILSEC:       Set flag BYTE (1 msec)
               RET
```

Timer generates an interrupt every 1 msec, but only 100 msec and 1 sec are wired to CPU RST 5.5.

OUTCAS:

 If test mask

Then:

 If OPPNTS (3) > = 100; OPPNTS (3) = 0, RET

 If OPPNTS (2) > 0, then

 Call cassette output

 OPPNTS (2) = OPPNTS (2) -1

 OPPNTS (3) = OPPNTS (3) +1

 END IF

Else:

 If OPPNTS (3) > = 64

 OPPNTS (3) = 111

 RET

 END

AIOCAS:

 Get ARGS + ADDRESSES

 Test if write; go to write

 Test if F.FORWARD

 ENT

Write:

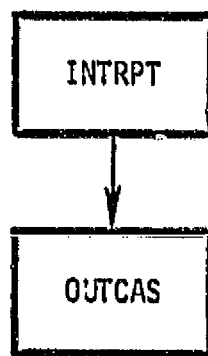
 Test status

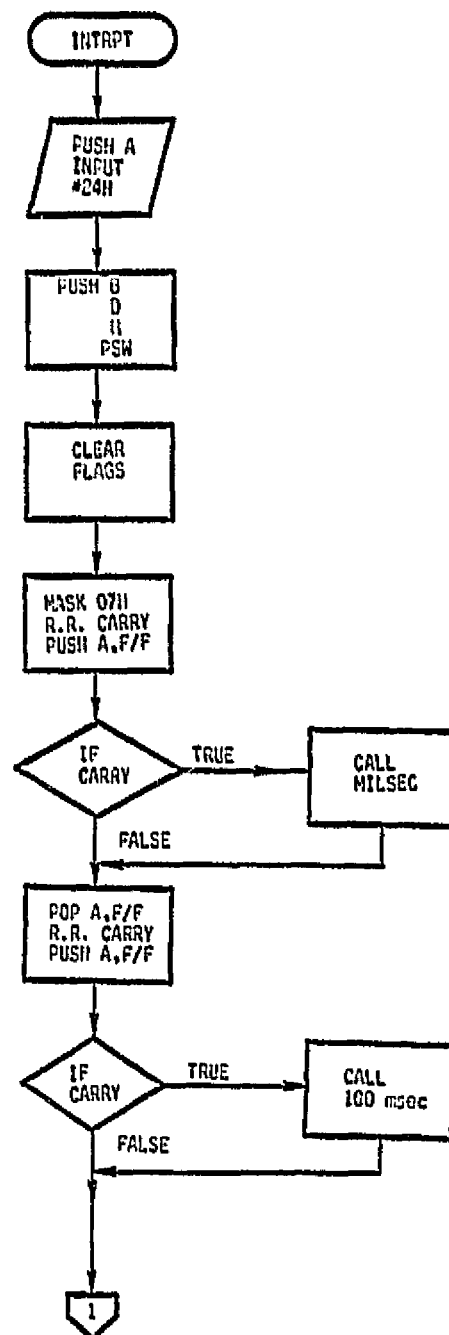
 Error go to error

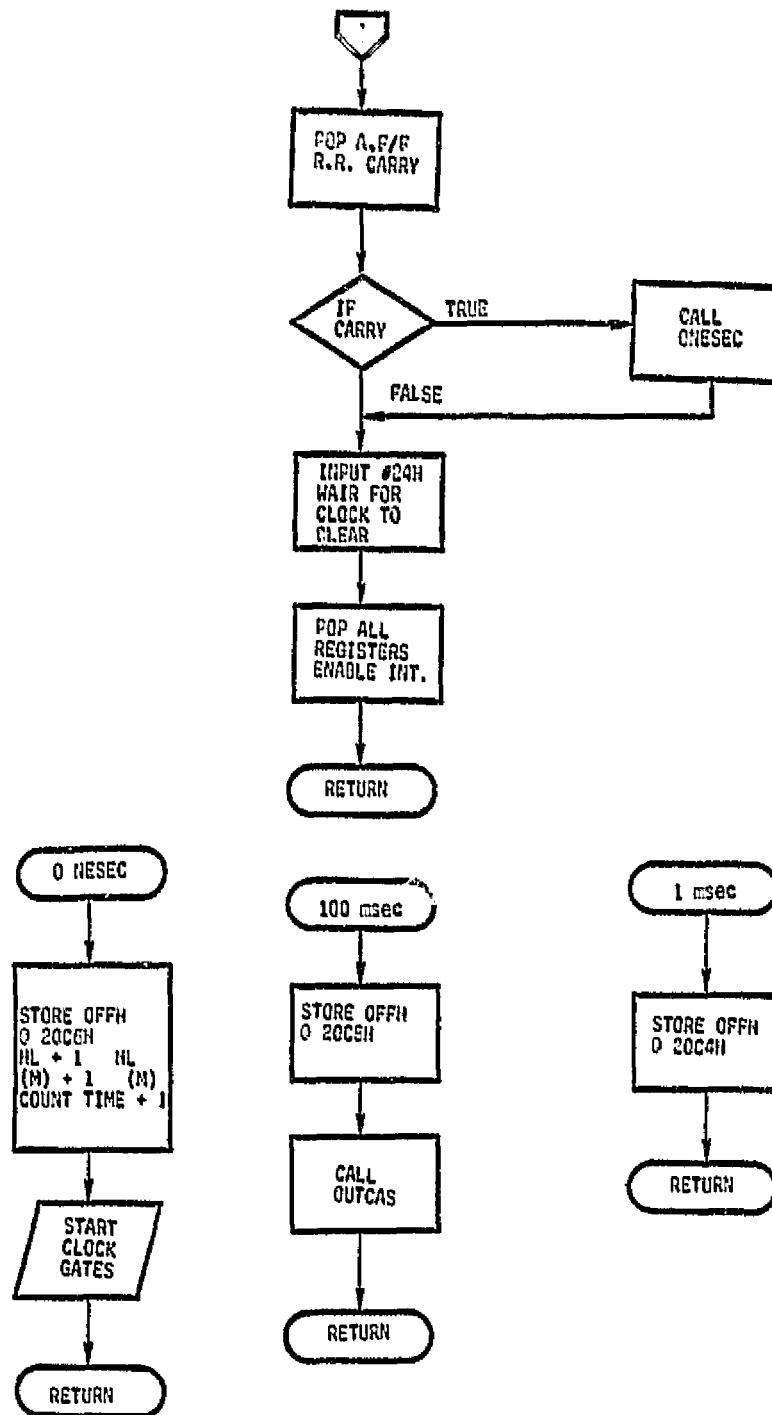
 Write 16 bits to ports

 Send write command 5 μ sec

 END







I/O UTILITY ROUTINES: INTERRUPT

Subroutine: INTREN

INTREN enables CPU interrupts.

Inputs: None

Outputs: None

Calls: Nothing

Destroys: Nothing

Modifies: CPU interrupt masks

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
Roger B. Fish

I/O UTILITY ROUTINES: INTERRUPT

Subroutine: INTRDS

INTRDS disables CPU interrupts.

Inputs: None

Outputs: None

Calls: Nothing

Destroys: Nothing

Modifies: CPU interrupt masks

Language: ASSEMBLER 8080/8085

L-V-A: Version 1, Level 1
 Roger B. Fish

APPENDIX B
PROGRAM LISTING

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT VCSFEG
 OBJECT MODULE PLACED IN :F1:VCSFEG.GPJ
 COMPILER INVOKED BY: FORT90 :F1:VCSFEG.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      PROGRAM VCSFEG
      C VCS MAIN SUPERVISOR PROGRAM
      C
      C MAIN PROGRAM CALLS SUPERVISOR ROUTINES BASED ON
      C FLAG AND OPERATION MODE CONTROLS
      C
      C INPUTS: CONTROL INFORMATION BUFFERS
      C OUTPUTS: NONE
      C CALLS: SUPINP, SUPOUT, SUPCNT, SUPDEG, SUPITZ,
      C         SUPERR, SUPEXT, SUPCAL, SUPHLT
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      $INCLUDE('F1:VCSMEN.FOR')
      = *****
      = C
      = C VCS SYSTEM MEMORY ALLOCATION
      = C
3      =      IMPLICIT INTEGER*(A-Z)
4      =      INTEGER*1  CID, KIB, DOB, TOB, PIB, POB, VOB
5      =      INTEGER*1  PKFEUF, PKREUF, PKFSRV, PKRSRV
6      =      INTEGER*1  NIXIPX(6), TFLAG(6)
7      =      INTEGER*1  ERREUF(8,10)
      = C
8      =      LOGICAL*1  IOFVEC(20), CNTVEC(16), SINT(8), SSK(8), DECM(8)
9      =      LOGICAL*1  DEFIOV(20), DEFCTV(16)
10     =      LOGICAL*1  NEXT, ADJUST, RESET, RUN, CALRT, FHRFAL, AUTO, MANUAL
      = C
11     =      CHARACTER*4  PARMES, MOSMES(10)
12     =      CHARACTER*1  DEPCOD(10)
13     =      CHARACTER*4  DOBC, DECN, CPMES(10)
14     =      CHARACTER*4  ERRMES(20), DECMES(20), RUNMES(20)
      = C
15     =      DIMENSION DEFRM(20), DEFPNT(8)
      = C
      = C COMMON BLOCK ALLOCATIONS
      = C
16     =      COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, FHRFAL, RUN, CALRT
      = C
17     =      COMMON/NESTB2/ PARMES(20)
      = C
18     =      COMMON/NESTB3/CPMES
      = C
19     =      COMMON/MOSMES/MOSMES
      = C
20     =      COMMON/SENPNT/SENPTR(16)
      = C
21     =      COMMON/PIFEUF/PKFEUF(10), PKREUF(10), PKFSRV, PKRSRV
      = C
22     =      COMMON/CLOCK/TFLAG

```

```

23 = C      COMMON/DEEUG/DEEUG
24 = C      COMMON/SENDEL/LODEL(40), FIDEL(60), FFEDEL(400)
25 = C      COMMON/IOFVEC/IOFVEC
26 = C      COMMON/INTVEC/INTVEC
27 = C      COMMON/OPSVEC/OPSVEC(10)
28 = C      COMMON/OFFPRM/ICHPNT(10), IMAX(16), IMXIPX
29 = C      COMMON/SENMEM/LSTPRS(3000), DECNIL(16)
30 = C      COMMON/OPSPNT/OPSPNTS(8)
31 = C      COMMON/WORK/TEMP10, DUMHY(19)
32 = C      COMMON/SAVER/SAVE(20)
33 = C      COMMON/PARAMS/  PARM(20)
34 = C      COMMON/ERRBUF/ERRBUF
35 = C      COMMON/ERRFLG/  ERROR(8)
36 = C      COMMON/COMBUF/CIB(80), KIB(8), DOB(8), TOB(120), PIB(2), FGB(2), VOB(2)
37 = C      COMMON/SENELF/SIB(10), SDB(10)
38 = C      COMMON/CONTAB/SENTAB(10, 2), RECDAT(10, 2)
39 = C      COMMON/NESTAB/ERRMES, DECMES, RUMMES, DSFCOD
40 = C      COMMON/DEFAULT/  DEF10V, DEFPRM, DEFCTV, DEFPNT
41 = C      COMMON/DEFTAB/  DEFRNG(10, 2), DEFCON(10, 2)
42 = C      COMMON/DEFOPS/DEFOPS(10)
43 = C      COMMON/DEFMOS/DEFMOS(16)
44 = C      SAVE /CPANEL/, /PIKEUF/, /SAVER/, /ERREUF/, /SENDEL/
45 = C      SAVE /DEEUG/, /IOFVEC/, /INTVEC/, /OPSVEC/
46 = C      SAVE /OFFPRM/, /SENMEM/, /OPSPNT/, /CLOCK/, /WORK/, /DEFTAB/
47 = C      SAVE /PARAMS/, /ERRFLG/, /COMBUF/, /SENBUF/, /CONTAB/, /NESTAB/, /DEFAULT/
48 = C      SAVE /SEIFNT/, /NESTB2/, /NESTB3/, /DEFOPS/
49 = C      SAVE /DEFMOS/
50 = C
51 = C      EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
52 = C      EQUIVALENCE (OPSVEC(1), OPNODE), (OPSVEC(2), MODECT), (OPSVEC(3), RMHMEK)
53 = C      EQUIVALENCE (OPSVEC(4), OPSENR), (OPSVEC(5), RATECK)
54 = C      EQUIVALENCE (OPSVEC(6), STATUS), (TIME, TFLAG(4))
55 = C
56 = C      EQUIVALENCE (IMAX(3), IMXSIR), (IMAX(2), IMXERR)

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

C
C*****
C
55 100 CONTINUE
C PROGRAM INITIALIZATION ENTRY POINT
C
56 CALL SUPITZ
C
57 CALL SUPCUT
C
C NORMAL OPERATION LOOP ENTRY POINT
C
58 1000 CONTINUE
C
C SENSOR INPUT CALL
C
59 CALL SUPINP
C
C TEST OPERATION MODE AND CALL SUPERVISOR ROUTINE
C
C MODE = 0 EXIT
60 IF(OPMODE EQ 0) CALL SUFEXT
C
C OPMODE = -1 ; RESTART
61 IF(OPMODE EQ -1) GO TO 100
C
C OPMODE = 1 ; NORMAL
62 IF(OPMODE EQ 1) CALL SUPCNT
C
C OPMODE = 2 ; DEBUG MODE
63 IF(OPMODE EQ 2) CALL SUFDEG
C
C OPMODE = 4 ; MANUAL MODE
64 IF(OPMODE EQ 4) CALL SUFCNT
C
C OPMODE = 5 ; CALIBRATE
65 IF(OPMODE EQ 5) CALL SUPCAL
C
C OPMODE = 3 ; HALT PROCESS
66 IF(OPMODE EQ 3) CALL SUFHLT
C
67 OFFINTS(7)=0
C
C BUFFER ERROR FLAGS FOR TEN SECONDS
C
68 DO 1010 I=L,NMXERR
69 ERREUF(1,MOD(TIME,10)+1)=ERROR(I)
70 1010 CONTINUE
C
C TEST FOR PRESENT ERROR AND PROCESS THEM
C
71 DO 1020 I=L,NMXERR
72 IF(ERROR(I).NE.0) THEN
73 TEMP10=I
C CALL ERROR PROCESSING ROUTINE
74 CALL SUFERR
75 END IF

```

ORIGINAL PAGE IS
OF POOR QUALITY

FORTRAN COMPILER

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C
      C IF MODE CHANGE SKIP OUT
76      IF (OPMODE.EQ.0) GO TO 1000
      C
77      1020 CONTINUE
      C
      C OUTPUT ROUTINE
      C
78      CALL SUPOUT
      C
79      GO TO 1000
      C
80      END

```

MODULE INFORMATION:

```

CODE AREA SIZE   = 0112H   2740
VARIABLE AREA SIZE = 0014H   200
MAXIMUM STACK SIZE = 0002H    20
178 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT VCSEPRG

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

1915-11 FORT90H-90 V2.1 COMPILATION OF PROGRAM UNIT SUPINP
 OBJECT MODULE PLACED IN :F1:VCSUP.OBJ
 COMPILER INVOKED BY: FORT90 :F1:VCSUP.FOR

```

1      SUBROUTINE SUPINP
      C
      C SUPINP CONTROLS INPUT TO THE VCS
      C
      C INPUTS: I/O CONTROL VECTORS
      C OUTPUTS: NOTHING
      C CALLS: SENINP, INPCDU, CONVRT
      C DESTROYS: NOTHING
      C MODIFIES: S, I, B, P, I, B.
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      LOGICAL*1 IOFVEC, IOFSEN
      C
4      COMMON/OPSVEC/OPMODE, ODUH(9)
      C
5      COMMON/IOFVEC/IOFVEC(20)
      C
6      EQUIVALENCE (IOFVEC(1), IOFSEN)
      C
7      SAVE /IOFVEC/, /OPSVEC/
      C
      C*****
      C
      C INPUT CONTROL PANEL INFORMATION
      C
8      CALL INPCDU
      C
      C EXIT IF HALT MODE
9      IF(OPMODE EQ 0) RETURN
      C
      C TEST SENSOR INPUT MASK AND CALL I/O ROUTINE IF TRUE
      C
10     IF(IOFSEN) THEN
      C
11     CALL SENINP
      C
12     CALL CONVRT
      C
13     END IF
      C
14     RETURN
15     END
  
```

ORIGINAL PAGE IS
 OF POOR QUALITY

MODULE INFORMATION:

CODE AREA SIZE = 001EH 300
 VARIABLE AREA SIZE = 0000H 00

FORTRAN CC.MYLER

PAGE 2

MAXIMUM STACK SIZE = 0002H 20
44 LINES READ
0 PROGRAM ERROR(S) IN PROGRAM UNIT SUPINP

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II FORTRAN-60 V2.1 COMPILATION OF PROGRAM UNIT SUPOUT
 OBJECT MODULE PLACED IN :F1:VCSUP.OBJ
 COMPILER INVOKED BY: FORT80 :F1:VCSUP FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE SUPOUT
      C
      C SUPOUT CONTROLS OUTPUTS FROM THE VCS
      C
      C INPUTS: I/O CONTROL VECTORS
      C OUTPUTS: NOTHING
      C CALLS: CNTOUT, OUTDSP, FORMAT, SAYDAT, INTREN, OUTPUT, CLOCK
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 TFLAG(6), BYTE, LEDS
4      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
5      CHARACTER*1 DSPCOD(18)
6      CHARACTER*4 DOBC, DOBN, ERRMES(20), DEGMES(20), RUNMES(20)
      C
7      LOGICAL*1 IOFVEC(20), IOFVOT, IOFCAS, CNTVEC(16), CHMPKF
      C
8      COMMON/OPSPNT/OPPNTS(8)
      C
9      COMMON/IOFVEC/IOFVEC
      C
10     COMMON/CLOCK/TFLAG
      C
11     COMMON/OPSVEC/OPMODE, MODECT, RANGCK, OPSENR, PATECK, OPSVEC(5)
      C
12     COMMON/MESTAB/ ERRMES, DEGMES, RUNMES, DSPCOD
      C
13     COMMON/CNTVEC/CNTVEC
      C
14     COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
15     EQUIVALENCE (BYTE, TEMP), (LEDS, XTEMP)
16     EQUIVALENCE (DOBX, DOB(5)), (CHMPKF, CNTVEC(3))
17     EQUIVALENCE (IOFVEC(15), IOFCAS), (IOFVEC(12), IOFVOT)
18     EQUIVALENCE (TIME, TFLAG(4))
19     EQUIVALENCE (DOBC, DOB(1)), (DOBN, DOB(5))
      C
20     SAVE /CLOCK/, /CNTVEC/
21     SAVE /OPSPNT/, /IOFVEC/, /MESTAB/, /COMBUF/, /OPSVEC/
      C
      C *****
      C
      C TEST OUTPUT CONTROL MASK AND CALL IF TRUE
      C
      C IF AUTO RUN OR MANUAL RUN THEN DO THIS
      C
22     IF(OPMODE.EQ.1.OR.OPMODE.EQ.4) THEN

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C
      C IF AUTO RUN
23      IF(OPMODE.EQ.1) THEN
24          CALL CNTOUT
25          DOBC=RUNMES(5)
      C
26      ELSE
      C ELSE MANUAL MODE
27          DOBC=DOBMES(17)
28      END IF
      C
      C PUT TIME INTO DISPLAY
29      DOBX=TIME
30      CALL FORMAT(DOEN)
      C
      C *RUNXXX OR OPERXXX
      C
31      CALL SAVDAT
32      CALL INTREN
33      GO TO 100
34      END IF
      C
      C DEBUG MODE
35      IF(OPMODE.EQ.2) THEN
36          DOBC=RUNMES(7)
37          DOBN=RLAMES(8)
      C
      C *DEBUG *
      C
38      GO TO 100
39      END IF
      C
      C CALIBRATE MODE
40      IF(OPMODE.EQ.5) THEN
41          DOBC=RUNMES(9)
42          DOBN=RUNMES(10)
43          C0B(1)=#2PH
      C
      C *CALBRAT
      C
44      END IF
      C
45      100 CONTINUE
      C
      C IF ERROR PUT IN DISPLAY
46      IF(OPPNTS(7).EQ.1) DOBN=RUNMES(15)
47      CALL OUTDSP
      C
      C BUILD LED DISPLAY BYTE
      C
48      IF(OPMODE.EQ.1) THEN
      C AUTO MODE
49          IOP=2
50          XTEMP=#02H
51      ELSE
52          IF(OPMODE.EQ.4) THEN
      C MANUAL MODE

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

52      IOP=1
53      XTEMP=#01H
54      C
55      ENDF
56      ENDF
57      C
58      TEMP=0
59      TEMP=TEMP+IOP*64
60      TEMP=TEMP+MODECT*16
61      TEMP=TEMP+RANGCK*8
62      TEMP=TEMP+OPSENR*4
63      C
64      C ERROR BIT
65      IF(OPPNTS(7).EQ.1) THEN
66          TEMP=TEMP+1
67          XTEMP=XTEMP+4
68      END IF
69      C
70      C PICK CONTROL
71      IF(CNMPICF) TEMP=TEMP+2
72      C
73      C OUTPUT STATUS TO LEDS
74      CALL OUTPUT(#000H,BYTE)
75      C
76      C OUTPUT STATUS TO LEDS
77      C INVERT BITS
78      XTEMP=-(XTEMP+1)
79      C XTEMP UPPER 4 BITS MUST BE ZEROS
80      XTEMP=(XTEMP.AND.#0FH)
81      CALL OUTPUT(#00H,LEDS)
82      C
83      C SYNCHRONIZE SYSTEM TIMING
84      CALL CLOCK
85      C
86      RETURN
87      END

```

MODULE INFORMATION:

```

CODE AREA SIZE      = 018CH   4440
VARIABLE AREA SIZE  = 0000H    60
MAXIMUM STACK SIZE  = 0000H   100
144 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SUPOUT

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT SUPCNT
OBJECT MODULE PLACED IN :F1:VCSUP.GBJ
COMPILER INVOKED BY: FORT90 :F1:VCSUP.FOR

```

1      SUBROUTINE SUPCNT
      C
      C SUPCNT CONTROLS VCS CONTROL ALGORITHM USE AND SIGNAL GENERATION
      C
      C INPUTS: OPERATION CONTROL VECTORS
      C OUTPUTS: NOTHING
      C CALLS: CNTRFD, CNTRRD
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      LOGICAL*1 CNTVEC(16), CNMOUT, CNMRER, CNMFRT
      C
4      COMMON/CNTVEC/CNTVEC
      C
5      EQUIVALENCE (CNTVEC(8), CNMFRT), (CNTVEC(9), CNMRER), (CNTVEC(10), CNMOUT)
      C
6      SAVE /CNTVEC/
      C
      C*****
      C
      C TEST FRONT CONTROL MASK
      C
7      IF(CNMFRT) CALL CNTRFD
      C
      C TEST REAR CONTROL MASK
      C
8      IF(CNMRER) CALL CNTRRD
      C
9      RETURN
10     END

```

MODULE INFORMATION:

CODE AREA SIZE = 0015H 21D
VARIABLE AREA SIZE = 0000H 00
MAXIMUM STACK SIZE = 0002H 2D
33 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT SUPCNT

1915-11 FORTRAN-90 V2.1 COMPILETIME OF PROGRAM UNIT SUPDBG
 OBJECT MODULE PLACED IN F1000H.VCSUP.OBJ
 COMPILER INVOKED BY: FORT90.F1000H FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

-      SUBROUTINE SUPDBG
      :
      : SUPDBG IS THE DEBUG OPERATION SUPERVISOR
      :
      : INPUTS: DEBUG CONTROL VECTOR
      : OUTPUTS: NONE
      : CALLS: DEBUGCI, DBUGCO, DEBUGIN, DEBUGOT, INTRO5
      :          DEBUGTP, DBUGIO, DBUGCK, DEBUGHE, ITZINP
      : TESTS: NOTHING
      : MODIFIES: NOTHING
      : VERSION 1.0
      :
      2      IMPLICIT INTEGER*2 (A-Z)
      :
      3      LOGICAL*1 (BOM*8)
      :
      4      COMMON/DEBUG/DBGM
      :
      5      COMMON/CPSEV1/CPMODE, MODECT, RANGCK, ARRY(7)
      :
      6      COMMON/WORK/TEMP10, DUMMY(19)
      :
      7      SAVE /DEBUG/, /CPSEV1/, /WORK/
      :
      :~~~~~
      : CALL INTERRUPT DISABLE
      8      CALL INTRO5
      :
      9      100 CONTINUE
      : INPUT DEBUG CONTROL FROM PANEL
      :
      10      CALL DEBUGCI
      :
      : TEST DEBUG MODE AND CALL CORRECT ROUTINE
      :
      11      IF (TEMP10.EQ.0) THEN
      : IF RESET SET TO EXIT MODE
      12      CPMODE=0
      13      RETURN
      14      END IF
      :
      15      IF (TEMP10.EQ.1) CALL DEBUGOT
      :
      16      IF (TEMP10.EQ.2) CALL DEBUGIN (0)
      :
      17      IF (TEMP10.EQ.3) CALL DEBUGIN (1)
      :
      18      IF (TEMP10.EQ.4) CALL ITZINP
      :
      19      IF (TEMP10.EQ.5) CALL DEBUGTP
  
```

PROGRAM FILED

ORIGINAL PAGE IS
OF POOR QUALITY

```

10      IF TEMP10.EQ.50 CALL DEVSVE
11      IF TEMP10.EQ.70 CALL DEVSVE
12      IF TEMP10.EQ.90 CALL DEVSVE
13      WRITE CONTROL PANEL AND WAIT FOR CONTINUE INPUT
14      CALL DEVSVE
15      GO TO 100
16      END

```

FILE INFORMATION

CODE AREA SIZE = 0000H 1600
 VARIABLE AREA SIZE = 0002H 20
 MAXIMUM STACK SIZE = 0002H 20
 64 LINES READ

0 PROGRAM ERRORS IN PROGRAM UNIT SUP060

ORIGINAL PAGE IS
OF POOR QUALITY

```

14 = : COMMON/SENDEL/ L1DEL(40), P0DEL(60), P01DEL(400)
15 = :
16 = : COMMON/IOFVEC/ IOFVEC
17 = :
18 = : COMMON/CONTVEC/ CONTVEC
19 = :
20 = : COMMON/OPSVEC/OPSVEC(10)
21 = :
22 = :
23 = : COMMON/OPPARM/ICMPNT(10), NMAX(15), NMXIPX
24 = :
25 = : COMMON/SENMEM/LSTPRS(3000), DSCNTL(16)
26 = :
27 = : COMMON/OPSPNT/OPPNTS(8)
28 = :
29 = : COMMON/WORK/TEMP10, DUMY(19)
30 = :
31 = : COMMON/SAVER/SAVE(20)
32 = :
33 = : COMMON/PARAMS/ PARAM(20)
34 = :
35 = : COMMON/ERRBUF/ERRBUF
36 = :
37 = : COMMON/ERRFLG/ ERROR(8)
38 = :
39 = : COMMON/COMBUF/CIB(50), KIB(8), DIB(8), TIB(128), PIB(2), PDB(2), WIS(2)
40 = :
41 = : COMMON/SENBUF/SIB(10), SDB(10)
42 = :
43 = : COMMON/CONTAB/SENTAB(10,2), PGCNT(10,2)
44 = :
45 = : COMMON/NESTAB/ERRNES, DBGNES, RUNNES, DSPCDD
46 = :
47 = : COMMON/DEFAULT/ DEFT0V, DEFFRM, DEFCV, DEFPNT
48 = :
49 = : COMMON/DEFTAB/ DEFRNG(10,2), DEFCN(10,2)
50 = :
51 = : COMMON/DEFOPS/DEFOPS(10)
52 = :
53 = : COMMON/DEFPOS/DEFPOS(16)
54 = :
55 = : SAVE /CPANEL/, /PIKBUF/, /SAVER/, /ERRBUF/, /SENDEL/
56 = : SAVE /DEBUG/, /IOFVEC/, /CONTVEC/, /OPSVEC/
57 = : SAVE /OPPARM/, /SENMEM/, /OPSPNT/, /CLOCK/, /WORK/, /DEFTAB/
58 = : SAVE /PARAMS/, /ERRFLG/, /COMBUF/, /SENBUF/, /CONTAB/, /NESTAB/, /DEFAULT/
59 = : SAVE /SENPT/, /NESTB2/, /NESTB3/, /DEFOPS/
60 = : SAVE /DEFPOS/
61 = :
62 = : EQUIVALENCE (DOB(1), D0EC), (DOB(5), D0EN)
63 = : EQUIVALENCE (OPSVEC(1), OPMODE), (OPSVEC(2), MODECT), (OPSVEC(3), PMODES)
64 = : EQUIVALENCE (OPSVEC(4), OPSERR), (OPSVEC(5), RATECK)
65 = : EQUIVALENCE (OPSVEC(6), STATUS), (TIME, TFLAG(4))
66 = :
67 = :
68 = : EQUIVALENCE (NMXSIR, NMAX(3)), (NMXERR, NMAX(2))
69 = : 1. (NMXIOF, NMAX(7)), (NMXFRM, NMAX(13)), (NMXCTV, NMAX(12)),
70 = : 2. (NMODES, NMAX(9)), (NMXPNT, NMAX(15)), (NMXSEN, NMAX(1)),

```


[illegible]

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C
      C SET DEFAULT I/O MODES
      C
      94      DO 300 I=L,NMIOF
      97      IOFVEC(I)=DEFIOV(I)
      98      300 CONTINUE
      C
      C SET DEFAULT PARAMETERS
      C
      89      DO 400 I=L,NMOPRM
      90      PARPRM(I)=DETPRM(I)
      91      400 CONTINUE
      C
      C SET DEFAULT CONTROL VECTOR
      C
      92      DO 500 I=L,NMXCTV
      93      CNTVEC(I)=DEFCVT(I)
      94      500 CONTINUE
      C
      95      OPPTS(1)=1
      C
      96      DO 1400 I=L,NMXSEN
      97      DO 1400 J=L,2
      98      RSCCAT(I,J)=DEFPNG(I,J)
      99      SENTAS(I,J)=DEFCON(I,J)
      100      1400 CONTINUE
      C
      C INITIALIZE THE OPERATION VECTOR
      C
      101      DO 1500 I=L,NMXOPS
      102      OPSVEC(I)=DEFOPS(I)
      103      1500 CONTINUE
      C
      C SET CMOS CONTROL
      C
      104      DO 1600 I=1,16
      105      DSCNTL(I)=DEFMOS(I)
      106      1600 CONTINUE
      C
      C USE RESET CODE TO CLEAR OTHER ARRAYS
      107      CALL RESET
      C
      C DISPLAY TAPE MESSAGE
      108      DGBG=DBGMES(12)
      109      DGBH=DBGMES(6)
      C @SYSTAP?
      C
      110      CALL PROMPT
      C TEST IF BLOCK FORMAT PAUSE IF .NOT. RUN MODE
      C SKIP IF RESET
      111      IF(RESET) GO TO 3000
      C IF CASSETTE ON, WRITE HEADER FILE
      C
      112      CALL TAPITZ
      C
      C CALL INTERACTIVE INITIALIZATION ROUTINE
      C

```

```
113 3000 CONTINUE
114      CALL ITZINP
      C
115 4000 CONTINUE
      C
116      RETURN
117      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

MODULE INFORMATION:

```
CODE AREA SIZE = 034CH 8440
VARIABLE AREA SIZE = 0016H 220
MAXIMUM STACK SIZE = 0006H 100
229 LINES READ
```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SUPITZ

ISIS-II FORTRAN 90 V2.1 COMPILATION OF PROGRAM UNIT SUPERR
 OBJECT MODULE PLACED IN :F1:V05SUP.OBJ
 COMPILER INVOKED BY: FORT90 :F1:V05SUP.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE SUPERR
      C
      C SUPERR PROCESSES ERROR CONDITION FLAGS AND OUTPUTS MESSAGE
      C
      C INPUTS: ERROR FLAG VECTOR
      C OUTPUTS: ERROR MESSAGE ON DISPLAY
      C CALLS: OUTDSP, INPCOU, FORMAT
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 ERRBUF(8,10)
4      INTEGER*1 COB, KIB, DOB, TOB, PIB, POB, VOB
      C
5      CHARACTER*1 DESPCOD(10)
6      CHARACTER*4 DOBC
7      CHARACTER*4 ERRMES(20), DGBMES(20), RUNMES(20)
      C
8      COMMON/MESTAB/ERR: 'S', DGBMES, RUNMES, DESPCOD
      C
9      COMMON/COMBUF/COB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
10     COMMON/ERRBUF/ERRBUF
      C
11     COMMON/PARAMS/PARAM(20)
      C
12     COMMON/OPSVEC/OPMODE, ODUM(9)
      C
13     COMMON/ERRFLG/ERROR(8)
      C
14     COMMON/OPSPNT/OPPNTS(8)
      C
15     COMMON/WORK/TEMP10, DUMMY(19)
      C
16     SAVE /OPSPNT/, /ERRFLG/, /OPSVEC/, /PARAMS/, /ERRBUF/
17     SAVE /MESTAB/, /COMBUF/, /WORK/
      C
18     EQUIVALENCE (PARAM(2), EARLMT)
19     EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
      C
      C*****
      C
      C LOAD ERROR MESSAGE TO DISPLAY BUFFER
      C
      C ERR
20     DOBC=RUNMES(15)
      C
      C PUT ERROR NUMBER IN DOBN
      C

```

```

21      TEMPX=TEMP10
22      COSH=135*TEMP10+ERRCOR(TEMP10)
      C
      C  FORMAT AND DISPLAY
23      CALL FORMAT(DDEN)
24      DOB(5)=DOB(6)
25      DOB(6)=#ICH
      C ERR H=XX ON DISPLAY
26      CALL OUTDSP
      C
      C  TEST FOR ERROR OVERFLOW
27      ETOT=0
28      DO 100 I=1,10
29      ETOT=ETOT+MIN0(1,ERRBUF(TEMPX,I))
30  100  CONTINUE
      C
      C  TEST IF ERROR COUNT OVER LIMIT
31      IF(ETOT GT ERRLMT) THEN
32      IF(OPMODE.EQ.1) OPMODE=4
33      END IF
      C
      C  PAUSE IF CAL ; ELSE RUN NO PAUSE
34      CALL INFCOU
      C
      C  SET ERROR FLAG
35      OPFNIS(7)=1
36      ERROR(TEMPX)=0
      C
37      RETURN
38      END

```

ORIGINAL PAGE IS
OF POOR QUALITY

MODULE INFORMATION:

```

CODE AREA SIZE      = 006AH    106D
VARIABLE AREA SIZE  = 0006H     6D
MAXIMUM STACK SIZE  = 000AH    10D
81 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SUPER

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT SUFEXT
OBJECT MODULE PLACED IN: FL:VCSUP.OBJ
COMPILER INVOKED BY: FORTR90:FL:VCSUP.FOR

```

1      SUBROUTINE SUFEXT
      C
      C SUFEXT STOPS THE VCS PROGRAM AFTER CLOSING DATA BASE
      C AND WAITS FOR OPERATOR INPUT AT THE C/D
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: OUTDSP, INTADS, CLOCK, INPCDU, INTREN, CNTOUT
      C DESTROYS: SIB, SOB, ERRORS
      C MODIFIES: CPMODE
      C VERSION 1.0
      C
2      INCLUDE('FL:VCSMEM.FOR')
      = C*****
      = C
      = C VCS SYSTEM MEMORY ALLOCATION
      = C
3      = IMPLICIT INTEGER*2(A-Z)
4      = INTEGER*4 CIO, KIO, DOB, TOB, PIO, POB, VO2
5      = INTEGER*4 PKFBUF, PKRBUF, PKFSAV, PKRSAY
6      = INTEGER*4 NKXIPX(6), TFLAG(6)
7      = INTEGER*4 EKRBUF(8,10)
      = C
8      = LOGICAL*4 IOFVEC(20), CNTVEC(16), SINT(8), SMSK(8), DEGM(8)
9      = LOGICAL*4 DEFION(20), DEFCTY(16)
10     = LOGICAL*4 NEXT, ADJUST, RESET, RUN, CALBRT, PHRFAL, AUTO, MANUAL
      = C
11     = CHARACTER*4 PARNES, NOSMES(10)
12     = CHARACTER*4 DSPC00(10)
13     = CHARACTER*4 DOBC, DOBN, OPSMES(10)
14     = CHARACTER*4 ERRMES(20), DEGMES(20), RUNMES(20)
      = C
15     = DIMENSION DEFRM(20), DEFPNT(8)
      = C
      = C COMMON BLOCK ALLOCATIONS
      = C
16     = COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALBRT
      = C
17     = COMMON/NEST62/ PARNES(20)
      = C
18     = COMMON/NEST63/ OPSMES
      = C
19     = COMMON/NOSMES/ NOSMES
      = C
20     = COMMON/SENPNT/SENPTR(16)
      = C
21     = COMMON/PIKBUF, PKRBUF(10), FKRBUF(10), PKFSAY, PKRSAY
      = C
22     = COMMON/CLOCK/ TFLAG
      = C
23     = COMMON/DEBUG/ DEGM

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      = C
24 = COMMON/SENDEL/LODEL(40), FODEL(60), FODEL(400)
      = C
25 = COMMON/IOFVEC/IOFVEC
      = C
26 = COMMON/CNTVEC/CNTVEC
      = C
27 = COMMON/OPSVEC/OPSVEC(10)
      = C
28 = COMMON/OPPARM/ICHPNT(10), NMPX(16), NMXIPX
      = C
29 = COMMON/SENMEM/LSTPAS(3000), DSCNTL(16)
      = C
30 = COMMON/OPSPNT/OPSPNTS(8)
      = C
31 = COMMON/WORK/TEMP10, DUMMY(19)
      = C
32 = COMMON/SAVER/SAVE(20)
      = C
33 = COMMON/PARAMS/ PARAM(20)
      = C
34 = COMMON/ERRBUF/ERRBUF
      = C
35 = COMMON/ERRFLG/ ERROR(8)
      = C
36 = COMMON/COMBUF/CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), WOB(2)
      = C
37 = COMMON/SENBUF/SIB(10), SDB(10)
      = C
38 = COMMON/CONTAB/SENTAB(10, 2), RGCDAT(10, 2)
      = C
39 = COMMON/NESTAB/ERRNES, DEGMES, RUNMES, DSPCOD
      = C
40 = COMMON/DEFAULT/ DEFIOV, DEFPRM, DEFCTV, DEFINT
      = C
41 = COMMON/DEFTAB/ DEFRNG(10, 2), DEFCON(10, 2)
      = C
42 = COMMON/DEFOPS/DEFOPS(10)
      = C
43 = COMMON/DEFMOS/DEFMOS(16)
      = C
44 = SAVE /CPANEL/, /PIKBUF/, /SAVER/, /ERRBUF/, /SENDEL/
45 = SAVE /DEBUG/, /IOFVEC/, /CNTVEC/, /OPSVEC/
46 = SAVE /OPPARM/, /SENMEM/, /OPSPNT/, /CLOCK/, /WORK/, /DEFTAB/
47 = SAVE /PARAMS/, /ERRFLG/, /COMBUF/, /SENBUF/, /CONTAB/, /NESTAB/, /DEFAULT/
48 = SAVE /SENINT/, /NESTB2/, /NESTB3/, /DEFOPS/
49 = SAVE /DEFMOS/
      = C
50 = EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
51 = EQUIVALENCE (OPSVEC(1), OPMODE), (OPSVEC(2), MODECT), (OPSVEC(3), PNGMSK)
52 = EQUIVALENCE (OPSVEC(4), OPSENR), (OPSVEC(5), RATECK)
53 = EQUIVALENCE (OPSVEC(6), STATUS), (TIME, TFLAG(4))
      = C
54 = INTEGER*1 BITS(8)
      = C
55 = LOGICAL*1 IOFCAS
      = C

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

56      EQUIVALENCE (IOFVEC(15), IOFCAS)
57      EQUIVALENCE (TEMP10,BITS(1))
58      EQUIVALENCE (NMAX(1),NMAXEN), (NMAXEP,NMAX(2))
      C
      C *****
      C
59      DOBC=RUNMES(19)
60      DOBN=RUNMES(19)
61      CALL OUTDSP
      C %-EXIT-%
      C
      C CLEAR VALVE OUTPUTS
62      VOB(1)=0
63      VOB(2)=0
      C
64      CALL CNTOUT
      C
65      TFLAG(2)=0
      C WAIT FOR CASSETTE TO FINISH IF ON
66      IF(IOFCAS) THEN
67        CALL INTREN
68      10  CALL CLOCK
      C WAIT FOR END OF BLOCK
69      END IF
      C
      C DISABLE INTERRUPTS
      C
70      CALL INTFDS
      C
      C CLEAR ERRORS
      C
71      DO 500 I=1,8
72      DO 600 J=1,10
73      ERRSUF(I,J)=0
74      600  CONTINUE
75      ERROR(I)=0
76      500  CONTINUE
      C
      C LOAD STOP MESSAGE INTO DOB
77      DOBC=RUNMES(16)
78      DOBN=RUNMES(17)
      C %-STOP-%
      C
79      CALL OUTDSP
      C
      C LOOP ON CONTROL PANEL TILL ENTRY BY OPERATOR
80      1000 CONTINUE
81      CALL INPCDU
      C
      C IF CALIBRATE ON THEN TEST NEXT AND ADJUST
82      IF(CALAPT) THEN
      C
      C IF RESET REBOOT SYSTEM
83      IF(RESET) THEN
84      OPMODE=-1
85      GO TO 2000
86      END IF

```


ORIGINAL PAGE IS
OF POOR QUALITY.

```

      C
      C IF NEXT GO TO DEBUG
97      IF(NEXT) THEN
98          OPMODE=2
99          GO TO 2000
99      END IF
      C
      C IF ADJUST GO TO CALIBRATE
91      IF(ADJUST) THEN
92          OPMODE=5
93          GO TO 2000
94      END IF
      C
      C END CALIBRATE SWITCH ON SECTION
95      END IF
      C
      C IF RUN SWITCH ON
      C RESTART RUN MODE ,CONTINUE
96      IF(RUN) THEN
      C
      C IF MANUAL SWITCH
97          IF(MANUAL) THEN
98              OPMODE=4
99              GO TO 2000
100         END IF
      C
      C IF AUTOMATIC SWITCH
101      IF(AUTO) THEN
102          OPMODE=1
103          GO TO 2000
104      END IF
      C
      C END RUN SWITCH ON SECTION
105      END IF
106      GO TO 1000
      C
107      2000 CONTINUE
108      RETURN
109      END

```

MODULE INFORMATION:

```

CODE AREA SIZE      = 0130H    3040
VARIABLE AREA SIZE = 0014H     200
MAXIMUM STACK SIZE = 000AH     100
284 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SUFEXT

ORIGINAL PAGE IS
OF POOR QUALITY

IS16-11 FORTAN-50 V2.1 COMPILATION OF PROGRAM UNIT SUPCAL
OBJECT MODULE PLACED IN :F1:VCSUP.OBJ
COMPILER INVOKED BY: FORT80 :F1:VCSUP.FOR

```

1      SUBROUTINE SUPCAL
      C
      C SUPCAL IS THE CALIBRATION ROUTINE SUPERVISOR
      C
      C INPUTS: NONE
      C OUTPUTS: OPMODE AND RESET SYSTEM (HOT RESTART)
      C CALLS: RESTRT, PROMPT
      C DESTROYS: BUFFERS FOR DELAYS AND POINTERS
      C MODIFIES: OPMODE
      C VERSION: 1.0
      C
2      $INCLUDE(:F1:VCSMEM.FOR)
      = C*****
      = C
      = C VCS SYSTEM MEMORY ALLOCATION
      = C
3      =      IMPLICIT INTEGER*2(A-Z)
4      =      INTEGER*1  CIB, KIB, DOB, TOS, PIB, POB, VOB
5      =      INTEGER*1  PKFBUF, PKRBUF, PKFSAY, PKRSAY
6      =      INTEGER*1  MAXIPX(6), TFLAG(6)
7      =      INTEGER*1  ERRBUF(3,10)
      = C
8      =      LOGICAL*1  IOFVED(20), CNTVED(16), SINT(8), SMSK(8), DEGM(8)
9      =      LOGICAL*1  DEFIOV(20), DEFCTV(16)
10     =      LOGICAL*1  NEXT, ADJUST, RESET, RUN, CALERT, PHRFAL, AUTO, MANUAL
      = C
11     =      CHARACTER*4  PARMES, MOSMES(10)
12     =      CHARACTER*1  DSPCOD(10)
13     =      CHARACTER*4  DOPC, DOBN, OPSMES(10)
14     =      CHARACTER*4  ERRMES(20), DEGMES(20), RUNMES(20)
      = C
15     =      DIMENSION DEFPNM(20), DEFPNT(8)
      = C
      = C COMMON BLOCK ALLOCATIONS
      = C
16     =      COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALERT
      = C
17     =      COMMON/MESTB2/ PARMES(20)
      = C
18     =      COMMON/MESTB3/OPSMES
      = C
19     =      COMMON/MOSMES/MOSMES
      = C
20     =      COMMON/SENPNT/SENPNT(16)
      = C
21     =      COMMON/PIKBUF/PKFBUF(10), PKRBUF(10), PKFSAY, PKRSAY
      = C
22     =      COMMON/CLOCK/TFLAG
      = C
23     =      COMMON/DEBUG/DEGM
      = C

```

```

24 = COMMON/SENDEL/LODEL(40), PODEL(60), FODEL(400)
   = C
25 = COMMON/IOFVEC/IOFVEC
   = C
26 = COMMON/CNTVEC/CNTVEC
   = C
27 = COMMON/OPSVEC/OPSVEC(10)
   = C
28 = COMMON/OPPARM/ICNPNT(10), NMAX(16), NMXPX
   = C
29 = COMMON/SENMEM/LSTFAS(3000), DSCNTL(16)
   = C
30 = COMMON/OPSPNT/OPSPNTS(8)
   = C
31 = COMMON/WORK/TEMP10, DUMMY(19)
   = C
32 = COMMON/SAVER/SAVE(20)
   = C
33 = COMMON/PARAMS/ PARAN(20)
   = C
34 = COMMON/ERRBUF/ERRBUF
   = C
35 = COMMON/ERRFLG/ ERROR(8)
   = C
36 = COMMON/COMBUF/CIB(80), KIB(8), DOB(8), TOB(128), FIB(2), POB(2), VOB(2)
   = C
37 = COMMON/SENBUF/SIB(10), SDB(10)
   = C
38 = COMMON/CONTAB/SENTAB(10, 2), RGCDAT(10, 2)
   = C
39 = COMMON/NESTAB/ERRNES, DBGNES, RUNNES, DEPCOD
   = C
40 = COMMON/DEFAULT/ DEF10Y, DEFPRH, DEFCTV, DEFNT
   = C
41 = COMMON/DEFTAB/ DEFANG(10, 2), DEFCON(10, 2)
   = C
42 = COMMON/DEFOPS/DEFOPS(10)
   = C
43 = COMMON/DEFMOS/DEFMOS(16)
   = C
44 = SAVE /CPANEL/, /PIKBUF/, /SAVER/, /ERRBUF/, /SENDEL/
45 = SAVE /DEBUG/, /IOFVEC/, /CNTVEC/, /OPSVEC/
46 = SAVE /OPPARM/, /SENMEM/, /OPSPNT/, /CLOCK/, /WORK/, /DEFTAB/
47 = SAVE /PARAMS/, /ERRFLG/, /COMBUF/, /SENBUF/, /CONTAB/, /NESTAB/, /DEFAULT/
48 = SAVE /SENPMI/, /NESTB2/, /NESTB3/, /DEFOPS/
49 = SAVE /DEFMOS/
   = C
50 = EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
51 = EQUIVALENCE (OPSVEC(1), OPNODE), (OPSVEC(2), MODECT), (OPSVEC(3), PNGMSK)
52 = EQUIVALENCE (OPSVEC(4), OPSENR), (OPSVEC(5), RATECK)
53 = EQUIVALENCE (OPSVEC(6), STATUS), (TIME, TFLAG(4))
   = C
C*****
C
54 DOBC=RUNNES(5)
55 DOBN=DBGNES(5)
C *UNITZ?

```

ORIGINAL PAGE IS
OF POOR QUALITY

FORTRAN COMPILER

ORIGINAL PAGE IS
OF POOR QUALITY

```

55      CALL FROMPT
56      C
57      IF (ADJUST) THEN
58      C CLEAR WORK VARIABLES FOR RESTART
59      CALL RESTRT
60      END IF
61      C
62      RETURN
63      END

```

MODULE INFORMATION:

```

CODE AREA SIZE      = 003AH    580
VARIABLE AREA SIZE = 0010H    160
MAXIMUM STACK SIZE = 0000H    100
118 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SUPCAL

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT SUPHLT
 OBJECT MODULE PLACED IN :F1:VOSUP.OBJ
 COMPILER INVOKED BY: FORT90 :F1:VOSUP.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE SUPHLT
      C
      C SUPHLT WILL HALT EXECUTION IN A RECOVERABLE WAY
      C
      C INPUTS: NONE
      C OUTPUTS: MODE STATUS
      C CALLS: INTREN, INTRDS, OUTDSP, INPCDU
      C DESTROYS: NOTHING
      C MODIFIES: OPMODE
      C VERSION 1.0
      C
      C*****
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*4 CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
4      LOGICAL*4 NEXT, ADJUST, RESET, RUN, CALENT, PHRFAL, AUTO, MANUAL
      C
5      CHARACTER*4 DOBC, DOBN, OPSMES(10)
6      CHARACTER*4 ERRMES(20), DBGMES(20), RUNMES(20)
7      CHARACTER*1 DSPCOD(10)
      C
8      COMMON/COMBUF/CIB(80), KIB(8), DOB(8), TOB(120), PIB(2), POB(2), VOB(2)
      C
9      COMMON/OPSVEC/OPMODE, ODUH(9)
      C
10     COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALENT
      C
11     COMMON/NESTAB/ERRMES, DBGMES, RUNMES, DSPCOD
      C
12     SAVE /OPSVEC/, /CPANEL/, /COMBUF/, /NESTAB/
      C
12     EQUIVALENCE (DOBC, DOB(1)), (DOBN, DOB(5))
      C
      C*****
      C
14     CALL INTRDS
      C
15     DOBC=RUNMES(11)
16     DOBN=RUNMES(12)
17     CALL OUTDSP
      C
18     10 CALL INPCDU
      C
      C TEST IF MODE SWITCH IS IN RUN
19     IF(RUN) THEN
      C
20     IF(ADJUST) THEN
      C IF RUN AND ADJUST GO TO MANUAL MODE
21     OPMODE=4
  
```

ORIGINAL PAGE IS
OF POOR QUALITY

```

22      GO TO 100
23      END IF
      C
24      IF (NEXT) GO TO 100
      C IF RUN MODE AND NEXT JUST CONTINUE WHERE LEFT OFF
      C
      C ELSE IT IS IN CALIBRATE POSITION
25      ELSE
26      IF (RESET) THEN
      C IF CALL MODE AND RESET STOP PROGRAM
27      OPMODE=0
28      GO TO 100
29      END IF
      C
30      END IF
      C WAIT
31      GO TO 10
      C
32  100  CALL INTREN
33      RETURN
34      END

```

MODULE INFORMATION:

```

CODE AREA SIZE   = 006AH   106D
VARIABLE AREA SIZE = 0028H   40D
MAXIMUM STACK SIZE = 000AH   10D
72 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SUFHLT

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ORIGINAL PAGE 18
OF POOR QUALITY

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT SENINP
OBJECT MODULE PLACED IN FILE WSEEM.OBJ
COMPILER INVOKED BY: FORTR90 FILE WSEEM.FOR

```

1      SUBROUTINE SENINP
      C
      C SENINP CALLS SENSOR INPUT ROUTINES UNDER I/O MASK CONTROL
      C
      C INPUTS: I/O MASK BUFFER
      C OUTPUTS: NONE
      C CALLS: INPCID, INPKS, AIADDC, SENRLN, ADCITZ
      C DESTROYS: NOTHING
      C MODIFIES: SENSOR INPUT BUFFER SIB
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-H, J-Z)
      C
3      IMPLICIT LOGICAL*1(I)
      C
4      DIMENSION DUMMY(5)
5      DIMENSION SIB(10), SDB(10)
      C
6      COMMON/IOFVEC/IOFVEC(5),
       1 IOFCID, IOFFKF, IOFFKR, IOFYCF, IOFYCR, IOFYOT, IOFDEP, IOFKEY,
       2 IOFCAS, IOFCRT, IOFCDU, IDUM(3)
      C
7      COMMON/OPSWEC/OPMODE, MODECT, RANGCK, OPSENR, RATECK, ODUM(5)
      C
8      COMMON/WORK/ TEMP10, SINPNT, WADUM(18)
      C
9      COMMON/SENSUF/ SIB, SDB
      C
10     SAVE /WORK/, /OPSWEC/
11     SAVE /IOFVEC/, /SENSUF/
      C
      C*****
      C INITIALIZE A/D EVERY TIME FOR SAFETYS SAKE
      C
12     CALL ADCITZ
      C
13     DO 100 J=1,5
      C TEST IF I/O FLAG ON (TRUE)
14       IF(IOFVEC(J+1)) THEN
15         SINPNT=J
      C CALL A/D DRIVER ROUTINE
16         CALL AIADDC (SINPNT, TEMP10)
17         SIB(SINPNT)=TEMP10
18       END IF
19     100 CONTINUE
      C
      C
      C TEST I/O FLAG FOR CID
      C
20     IF(IOFCID) CALL INPCID
      C

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C TEST I/O FLAG FOR PICKS
      C
21      IF(IOFFNE OR IOFFER) CALL INFEES
      C
      C REFINAPE PATCH FOR A/D INPUT WIPES
22      DO 200 J=2,5
23      DUMMY(J)=SIB(J)
24      200 CONTINUE
      C
25      SIB(2)=DUMMY(4)
26      SIB(3)=DUMMY(5)
27      SIB(4)=DUMMY(2)
28      SIB(5)=DUMMY(3)
      C
      C IF RATE CHECKING ACTIVE CALL SENSOR LIMIT ROUTINE
      C
29      IF(RATECK.EQ.1) CALL SENRLM
      C
30      RETURN
31      END

```

MODULE INFORMATION:

```

CODE AREA SIZE      = 005EH    1900
VARIABLE AREA SIZE = 000CH    120
MAXIMUM STACK SIZE = 0002H     20
71 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SENINP

TS15-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT CONVRT
 OBJECT MODULE PLACED IN: F1:V05E01.OBJ
 COMPILER INVOKED BY: FORTR0 F1:V05E01.FCR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE CONVRT
      C
      C CONVRT CONVERTS SENSOR INPUT DATA TO ENGINEERING UNITS, AND
      C      DOES OPTIONAL RANGE CHECK
      C
      C INPUTS: SENSOR INPUT DATA IN SIB
      C OUTPUTS: SENSOR DATA IN SDB
      C CALLS: CONFMC
      C DESTROYS: NOTHING
      C MODIFIES: SDB
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-H, J-Z)
      C
3      INTEGER*4 NMXIPX
      C
4      DIMENSION SIB(10), SDB(10)
      C
5      LOGICAL*4 IOFVEC(20)
      C
6      COMMON/OPARM/ICNFNT(10), NMXVEC(16), NMXIPX(6)
      C
7      COMMON/IOFVEC/ IOFVEC
      C
8      COMMON/SENBUF/ SIB, SDB
      C
9      COMMON/WORK/ TEMP10, SINPNT, DUMMY(18)
      C
10     EQUIVALENCE (NMXSEN, NMXVEC(1))
      C
11     SAVE /IOFVEC/, /SENBUF/, /OPARM/, /WORK/
      C
      C*****
      C
12     DO 100 I=L, NMXSEN
      C TEST IF I/O FLAG ON (TRUE)
13     IF (IOFVEC(I+1)) THEN
14         SINPNT=I
15         TEMP10=SIB(I)
      C CALL DATA TO ENGINEERING UNITS CONVERSION
16         CALL CONFMC
17         SDB(I)=TEMP10
18     END IF
19     100 CONTINUE
      C
20     RETURN
21     END

```

MODULE INFORMATION:

CODE AREA SIZE = 0056H 66D
VARIABLE AREA SIZE = 0014H 40
MAXIMUM STACK SIZE = 0002H 2D
47 LINES READ

ORIGINAL PAGE IS
OF POOR QUALITY

0 PROGRAM ERROR(S) IN PROGRAM UNIT CONVEY

ISIS-11 FORTRAN-90 V2.1 COMPILE OF PROGRAM UNIT CNTOUT
 OBJECT MODULE PLACED IN: F140SEEM.OBJ
 COMPILER INVOKED BY: FORT90 F140SEEM.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE CNTOUT
      C
      C CNTOUT IS THE CONTROLLER OUTPUT ROUTINE. THIS CALLS THE
      C   APPROPRIATE OUTPUT ROUTINES AS PER THE I/O MASKS
      C
      C INPUTS: I/O MASKS
      C OUTPUTS: NONE
      C CALLS: AIOVOT
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-H, J-Z)
      C
3      IMPLICIT LOGICAL*4 (I)
      C
4      INTEGER*4 CIB(8), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
5      COMMON/COMBUF/ CIB, KIB, POB, TOB, PIB, POB, VOB
      C
6      COMMON/SENEUF/ SIB(10), SDB(10)
      C
7      COMMON/PARAMS/ DMAX, DMIN, PDUM(18)
      C
8      COMMON/IOFVEC/ IUM(9), IOFVCF, IOFVCR, IOFVOT, IBDUM(8)
      C
9      SAVE /IOFVEC/, /COMBUF/, /PARAMS/, /SENEUF/
      C
10     EQUIVALENCE (SDB(4), FDRUM), (SDB(5), RDRUM)
      C
      C *****
      C
      C VOB(1) = FRONT VALVE
      C VOB(2) = REAR VALVE
      C 1 = UP CONTROL
      C -1 = DOWN CONTROL
      C
      C OUTPUT FORMAT TO VALVE DRIVER
      C BIT 0 = FRONT UP
      C BIT 1 = FRONT DOWN
      C BIT 2 = REAR UP
      C BIT 3 = REAR DOWN
      C DRUMS ARE LIMITED TO DMAX UP AND DMIN DOWN
      C
11     CBYTE=0
      C FRONT VALVE ENABLE TEST
12     IF (IOFVCF) THEN
13     VTEST=0
14     VTEST=VOB(1)
      C TEST IF VALVE TO BE CONTROLLED
15     IF (VTEST.NE.0) THEN

```

FORTRAN COMPILER

ORIGINAL PAGE IS
OF POOR QUALITY

```

C TEST IF UP OR DOWN
16 IF(VTEST.EQ.1) THEN
17   IF(FORUM.LT.DMAX) CBYTE=1
18 ELSE
19   IF(FORUM.GT.DMIN) CBYTE=2
20 END IF
21 END IF
C END IOFVCF BLOCK
22 END IF
C
C REAR VALVE ENABLE
23 IF(IOFVCR) THEN
24   VTEST=3
25   VTEST=VDB(2)
C TEST IF VALVE TO BE CONTROLLED
26 IF(VTEST.NE.0) THEN
C
C TEST IF UP OR DOWN CONTROL
27 IF(VTEST.EQ.1) THEN
28   IF(RDFUL.LT.DMAX) CBYTE=CBYTE+4
29 ELSE
30   IF(RDFUL.GT.DMIN) CBYTE=CBYTE+8
31 END IF
32 END IF
C END IOFVCR BLOCK
33 END IF
C
C TEST IF VALVE OUTPUT ENABLED
34 IF(IOFVOT) THEN
35   CALL RIOWOT(CBYTE)
36 END IF
C
37 RETURN
38 END

```

MODULE INFORMATION:

```

CODE AREA SIZE   = 0000H   1920
VARIABLE AREA SIZE = 0004H   40
MAXIMUM STACK SIZE = 0002H   20
86 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT CNTAUT

ISIS-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT ITZINP
 OBJECT MODULE PLACED IN :F1:VCSSEM.OBJ
 COMPILER INVOKED BY: FORTE0 :F1:VCSSEM.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE ITZINP
      C
      C ITZINP PROMPTS THE USER AND UPDATES THE VALUE OF SOME VARIABLES IN COMMON
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: OUTDSP, INPCDU, INTRDS, INPHEX, OUTPUT
      C DESTROYS: NOTHING
      C MODIFIES: VARIABLES SELECTED BY USER
      C VERSION 1.0
      C
2      $INCLUDE(:F1:VCSSEM.FOR)
      = C *****
      = C
      = C VCS SYSTEM MEMORY ALLOCATION
      = C
3      =      IMPLICIT INTEGER*2(A-Z)
4      =      INTEGER*1  CIB, KIB, DOB, TOB, PIB, POB, VOB
5      =      INTEGER*1  PKFBUF, PKRBUF, PKFSAV, PKRSV
6      =      INTEGER*1  MNXIPX(6), TFLAG(6)
7      =      INTEGER*1  ERRBUF(8,10)
      = C
9      =      LOGICAL*1  IOFVEC(20), ONTVEC(16), SINT(8), SMEX(8), DBGM(8)
9      =      LOGICAL*1  DEFIOV(20), DEFCTV(16)
10     =      LOGICAL*1  NEXT, ADJUST, RESET, RUN, CALBRT, PWRPAL, AUTO, MANUAL
      = C
11     =      CHARACTER*4  PARMES, MOSMES(10)
12     =      CHARACTER*1  DSPC00(10)
13     =      CHARACTER*4  DGBC, DGBN, OPSMES(10)
14     =      CHARACTER*4  ERRMES(20), DBGMES(20), RUNMES(20)
      = C
15     =      DIMENSION DEFPRM(20), DEFPNT(8)
      = C
      = C COMMON BLOCK ALLOCATIONS
      = C
16     =      COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PWRPAL, RUN, CALBRT
      = C
17     =      COMMON/NESTB2/ PARMES(20)
      = C
18     =      COMMON/NESTB3/OPSMES
      = C
19     =      COMMON/MOSMES/MOSMES
      = C
20     =      COMMON/SENFNT/SENPTR(16)
      = C
21     =      COMMON/PIKBUF/PKFBUF(10), PKRBUF(10), PKFSAV, PKRSV
      = C
22     =      COMMON/CLOCK/TFLAG
      = C
23     =      COMMON/DEBUG/DBGM
      = C

```

```

24 = COMMON/SENDEL/LODEL(40), FODEL(60), FODEL(400)
   = C
25 = COMMON/IOFVEC/IOFVEC
   = C
26 = COMMON/CNTVEC/CNTVEC
   = C
27 = COMMON/OPSVEC/OPSVEC(10)
   = C
28 = COMMON/OPPARM/ICNPNT(10), NMAX(16), NMAXPX
   = C
29 = COMMON/SENMEM/LSTPAS(3000), DSCNTL(16)
   = C
30 = COMMON/OPSPNT/OPPNTS(8)
   = C
31 = COMMON/WORK/TEMP10, DUMMY(19)
   = C
32 = COMMON/SAVER/SAVE(20)
   = C
33 = COMMON/PARAMS/ PARAM(20)
   = C
34 = COMMON/ERRBUF/ERRBUF
   = C
35 = COMMON/ERRFLG/ ERROR(8)
   = C
36 = COMMON/COMBUF/CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), WOB(2)
   = C
37 = COMMON/SENBUFF/SIB(10), SOB(10)
   = C
38 = COMMON/CONTAB/SENTAB(10, 2), RGCDAT(10, 2)
   = C
39 = COMMON/NESTAB/ERRNES, DBGMES, RUNMES, DEFCOD
   = C
40 = COMMON/DEFAULT/ DEFION, DEFPRN, DEFCTV, DEFNT
   = C
41 = COMMON/DEFTAB/ DEFRNG(10, 2), DEFCON(10, 2)
   = C
42 = COMMON/DEFOPS/DEFOPS(10)
   = C
43 = COMMON/DEFMOS/DEFMOS(16)
   = C
44 = SAVE /CPANEL/, /PIVBUF/, /SAVER/, /ERRBUF/, /SENDEL/
45 = SAVE /DEBUI/, /IOFVEC/, /CNTVEC/, /OPSVEC/
46 = SAVE /OPPARM/, /SENMEM/, /OPSPNT/, /CLOCK/, /WORK/, /DEFTAB/
47 = SAVE /PARAMS/, /ERRFLG/, /COMBUF/, /SENBUFF/, /CONTAB/, /NESTAB/, /DEFAULT/
48 = SAVE /SENPNL/, /NESTB2/, /NESTB3/, /DEFOPS/
49 = SAVE /DEFMOS/
   = C
50 = EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
51 = EQUIVALENCE (OPSVEC(1), OPMODE), (OPSVEC(2), MODECT), (OPSVEC(3), PNGMSK)
52 = EQUIVALENCE (OPSVEC(4), OPSENR), (OPSVEC(5), RATECK)
53 = EQUIVALENCE (OPSVEC(6), STATUS), (TIME, TFLAG(4))
   = C
54 = INTEGER*1 BITS(8)
   C
55 = CHARACTER*1 CDUM
   C
56 = EQUIVALENCE (NMAXPX, NMAX(4))

```

ORIGINAL PAGE IS
OF FOUR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

```

57      EQUIVALENCE (TEMP10,BITS(1))
58      EQUIVALENCE (OPPNT5(1),DATPNT)
59      EQUIVALENCE (CDUM,DOB(1)),(DOB(5),DOBX)
      C
      C *****
      C
60      CALL INTRDS
      C
      C LOAD DOB WITH INITIALIZE CODE DISPLAY AND PAUSE
      C
61      DOBC=DEGMES(12)
62      DOBN=DEGMES(5)
      C @SYSITZ?
63      10  CALL OUTDSP
      C
64      CALL INPCDU
      C IF RUN SWITCH ON WAIT FOR RUN MODE FROM OPERATOR
      C BEGIN BLOCK = RUN
65      IF(RUN) THEN
      C TEST IF AUTOMATIC MODE SWITCH
66      IF(AUTO) THEN
67      OPMODE=1
68      GO TO 2000
69      END IF
      C TEST IF MANUAL MODE
70      IF(MANUAL) THEN
71      OPMODE=4
72      GO TO 2000
73      END IF
74      GO TO 10
75      END IF
      C END BLOCK = RUN
      C
      C ELSE CALIBRATE SWITCH
      C IF PST SKIP OUT OF ROUTINE
76      IF(RESET) GO TO 1000
      C
      C DATA INITIALIZATION BLOCK DO
      C BEGIN BLOCK = DATA
77      IF(ADJUST) THEN
      C
78      DO 100 I=L,NMXPIM
      C
      C LOAD DOB WITH PROMPT(I)
      C
      C LOAD MESSAGE FROM RUNMES TABLE TO CONTROL PROMP TYPES
      C
      C PROMPTS
      C ?I/OMASK
      C ?CHTMASK
      C ?OPSPARM
      C ?VRBPARM
      C DAT?ITZ?
      C ?SENPARM
      C ?MOSCHTL
      C
79      DOBC=RUNMES(I)

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

90      DOBN=RUNMES(13)
      C
91      IF(I.GE.3) DOBN=RUNMES(14)
      C
92      IF(I.EQ.5) THEN
93          DOEC=DEGMES(4)
94          DOBN=DEGMES(5)
95      END IF
      C
96      IF(I.EQ.6) THEN
97          DOBC=ERRMES(1)
98          DOBN=RUNMES(14)
99          CDUM=DSPCOD(7)
100      END IF
      C
101      IF(I.EQ.7) THEN
102          DOBC=DEGMES(3)
103          DOBN=RUNMES(2)
104      END IF
      C
      C DISPLAY PROMPT
      C
105      CALL OUTDSP
      C
      C READ INPUT FROM C/D PANEL
      C
106      CALL INPCDU
      C
      C TEST IF MODIFY THIS SET OF VARIABLES
      C BEGIN BLOCK = MODIFY
107      IF(ADJUST) THEN
      C MODIFY THIS SET
      C
108      IF(I.EQ.5) GO TO 300
      C
109      NMXIPT=NMXIPX(I)
110      DO 200 J=L,NMXIPT
111      TEMPIO=0
      C
      C LOAD DOB WITH PROMPT
      C
      C I/O FLAG RESET
112      IF(I.EQ.1) THEN
113          IF(10FVEC(J)) TEMPIO=TEMPIO+255
114          DOBC=ERRMES(J)
115          GO TO 150
116      END IF
      C
      C CONTROL VECTOR RESET
117      IF(I.EQ.2) THEN
118          DOBC=ERRMES(10NPNT(J))
119          IF(10FVEC(J)) TEMPIO=TEMPIO+255
120          GO TO 150
121      END IF
      C
      C OPERATION VECTOR RESET
122      IF(I.EQ.3) THEN

```



```

112      DOBC=OPSRES(J)
114      TEMP10=TEMP10+OPSVEC(J+1)
115      GO TO 150
116      END IF
      C
      C PARAMETER RESET
117      IF(I EQ 4) THEN
118      DOBC=PARRES(J)
119      TEMP10=TEMP10+PARAM(J)
120      GO TO 150
121      END IF
      C
      C INPUT TABLE RESET
      C BEGIN BLOCK = TABLE
122      IF(I EQ 6) THEN
123      K=(J-1)/2+1
124      L=MOD(J-1,2)
      C
125      IF(J LE 16) THEN
126      TEMP10=SENTAB(K,L+1)
127      ELSE
128      TEMP10=RGCCAT(K-8,L+1)
129      END IF
130      IF(J LE 16) THEN
131      DOBC=ERRRES(K+1)
132      COUN=DSPCOD(5)
      C #
133      IF(L EQ 1) COUN=DSPCOD(2)
      C *
134      END IF
      C
135      IF(J GE 17) THEN
136      DOBC=ERRRES(K-7)
      C NXXX
137      DOB(1)=#4EH
      C XXXX
138      IF(L EQ 1) DOB(1)=#58H
139      END IF
140      GO TO 150
      C END BLOCK = TABLE
141      END IF
      C
142      IF(I EQ 7) THEN
143      DOBC=MOSRES(J)
144      TEMP10=TEMP10+OSCNL(J)
145      GO TO 150
146      END IF
      C
      C ##
147      DOBX=TEMP10
148      CALL FORMAT(DGBX)
      C DISPLAY MESSAGE
      C
149      CALL OUTDSP
      C
150      CALL INPCDU
      C IF RST SKIP OUT OF THIS SET

```

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

```

151      IF(RESET) GO TO 100
      C
      C IF ADJ THEN DO . ELSE SKIP TO NEXT
      C BEGIN BLOCK = ADJUST
152      IF(ADJUST) THEN
153      CALL INP4EX
      C
      C BEGIN BLOCK = LOAD
154      IF(TEMP10.NE.0) THEN
      C LOAD DATA IN TEMP10+1 INTO DESTINATION
      C
      C I/O FLAG SET UP
      C
155      IF(I.EQ.1) THEN
156      IF(DUMMY(1).GE.1) THEN
157      IOFVEC(J)=.TRUE
158      ELSE
159      IOFVEC(J)=.FALSE
160      END IF
161      END IF
      C
      C CONTROL VECTOR SET UP
      C
162      IF(I.EQ.2) THEN
163      IF(DUMMY(1).GE.1) THEN
164      CNTVEC(J)=.TRUE
165      ELSE
166      CNTVEC(J)=.FALSE
167      END IF
168      END IF
      C
      C OPERATING CONTROL PARAMETER SET UP
      C
169      IF(I.EQ.3) OPSVEC(J+1)=DUMMY(1)
      C
      C PARAMETER SET UP
      C
170      IF(I.EQ.4) PARAM(J)=DUMMY(1)
      C
      C SENSOR TABLE RESET
      C
171      IF(I.EQ.6) THEN
      C
172      IF(J.LE.16) THEN
173      SENTAB(K,L+1)=DUMMY(1)
174      ELSE
175      RGCDAT(K-8,L+1)=DUMMY(1)
176      END IF
      C END SENSOR SET UP
177      END IF
      C
178      IF(I.EQ.7) DECHTL(J)=DUMMY(1)
      C
      C END OF DATA LOAD SECTION
179      END IF
      C END BLOCK = LOAD
      C

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C END OF LOOP FOR SET
190      END IF
      C END BLOCK = ADJUST
      C
191      200 CONTINUE
      C
      C END OF SET OF VARIABLES
192      END IF
      C END BLOCK = MODIFY
      C
193      GO TO 450
      C
      C CLEAR CMOS AND SET DEFAULTS
194      300 DOBN=RUNMES(20)
195      DOB(4)=#41H
196      CALL OUTDSP
      C
197      DO 400 I=L,1000
198      LSTRAS(I)=-32000
199      400 CONTINUE
      C
      C SET CMOS BUFFER POINTER AT START (OPPNTS(1))
199      DATPNT=1
      C
191      450 CONTINUE
      C
192      100 CONTINUE
      C
      C END OF DATA INITIALIZATION BLOCK
192      END IF
      C END BLOCK = DATA
      C
      C *****
      C
      C TEST FOR DEBUG OR CALIBRATE MODES
      C
194      DOBC=RUNMES(9)
195      DOBN=RUNMES(10)
      C %CALIBRAT
196      CALL OUTDSP
      C
197      CALL INPCDU
      C IF ADJ PRESSED
198      IF(ADJUST) THEN
199      OPMODE=5
200      GO TO 1000
201      END IF
      C IF RESET PRESSED
202      IF(RESET) GO TO 1000
      C *****
      C DEBUG
203      DOBC=RUNMES(7)
204      DOBN=DEGMES(12)
205      COUN=DSFCOD(7)
      C %DEBSTATS
206      CALL OUTDSP
      C

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

207      CALL INPCD0
      C IF A0J PRESSED THEN DO
208      IF(A0J) THEN
209      OFMODE=2
210      GO TO 1000
211      END IF
      C IF RESET SKIP OUT
212      IF(RESET) GO TO 1000
      C
213      1000 CONTINUE
214      RETURN
      C
      C SYNCHRONIZE CLOCK TIMER
215      2000 CONTINUE
216      CALL OUTPUT(125H,0)
217      CALL OUTPUT(125H,107H)
      C
218      RETURN
219      END

```

MODULE INFORMATION:

```

CODE AREA SIZE      = 062EH   1582D
VARIABLE AREA SIZE = 001EH    38D
MAXIMUM STACK SIZE = 0009H    10D
412 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT ITZ1NP

IS15-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT PESTRT
 OBJECT MODULE PLACED IN :F1:VCSSEMI.OBJ
 COMPILER INVOKED BY: FOR90 :F1:VCSSEMI.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY.

```

1      SUBROUTINE PESTRT
      C
      C RESTRT CLEARS APPROPRIATE DAT TO ALLOW WARM START OF VCS
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: NOTHING
      C DESTROYS: SENPNTS, ERREBUF, OPPNTS, CLOCK, PIKEBUF, SENDELS
      C MODIFIES: OPPNTS(1), STATUS
      C VERSION 1.0
      C
2      $INCLUDE('F1:VCSSEMI.FOR')
      C*****
      C
      C VCS SYSTEM MEMORY ALLOCATION
      C
3      =      IMPLICIT INTEGER*2(A-Z)
4      =      INTEGER*4  CIB, KIB, DOB, TOB, PIB, POB, VOB
5      =      INTEGER*4  PKFBUF, PKPBUF, PKFSAY, PKRSAY
6      =      INTEGER*4  NMKIPX(6), TFLAG(6)
7      =      INTEGER*4  ERREUF(8,10)
      =      C
8      =      LOGICAL*4  IOFVEC(20), CNTVEC(16), SINT(8), SMSK(8), DEGN(8)
9      =      LOGICAL*4  DEFTOV(20), DEFCTV(16)
10     =      LOGICAL*4  NEXT, ADJUST, RESET, RUN, CALBRT, PHRFAL, AUTO, MANUAL
      =      C
11     =      CHARACTER*4  PARKES, MOSMES(10)
12     =      CHARACTER*4  DSPCOD(10)
13     =      CHARACTER*4  DOEC, DOBN, OPSMES(10)
14     =      CHARACTER*4  ERRMES(20), DBGMES(20), RUNMES(20)
      =      C
15     =      DIMENSION DEFRM(20), DEFPH(8)
      =      C
      =      C COMMON BLOCK ALLOCATIONS
      =      C
16     =      COMMON/CPRMEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALBRT
      =      C
17     =      COMMON/PESTB2/ PARKES(20)
      =      C
18     =      COMMON/PESTB3/OPSMES
      =      C
19     =      COMMON/MOSMES/MOSMES
      =      C
20     =      COMMON/SENPNT/SENPTR(16)
      =      C
21     =      COMMON/PIKBUF/PKFBUF(10), PKRBUF(10), PKFSAY, PKRSAY
      =      C
22     =      COMMON/CLOCK/TFLAG
      =      C
23     =      COMMON/DEBUC/DBGM
      =      C

```

```

24 = COMMON/SENDEL/LODEL(40), PCDEL(20), FORDEL(400)
   = C
25 = COMMON/IOFVEC/IOFVEC
   = C
26 = COMMON/CNTVEC/CNTVEC
   = C
27 = COMMON/OPSVEC/OPSVEC(10)
   = C
29 = COMMON/OPPARM/ICHENT(10), NMAX(16), NMAXPX
   = C
29 = COMMON/SENHEN/LSTPAS(3000), DSCNTL(16)
   = C
30 = COMMON/OPSPNT/OPSPNTS(8)
   = C
31 = COMMON/WORK/TEMP10, DUMMY(19)
   = C
32 = COMMON/SAVER/SAVE(20)
   = C
33 = COMMON/PARAMS/ PARAM(20)
   = C
34 = COMMON/EPBBUF/EPBBUF
   = C
35 = COMMON/ERRFLG/ ERROR(8)
   = C
35 = COMMON/COMBUF/CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
   = C
37 = COMMON/SENBUFF/STB(10), SDB(10)
   = C
38 = COMMON/CONTAB/SENTAB(10, 2), RCCDAT(10, 2)
   = C
39 = COMMON/MESTAB/ERRMES, DBGMES, RUNMES, DEPCOD
   = C
40 = COMMON/DEFAULT/ DEFIOV, DEFPPN, DEFCTV, DEFINT
   = C
41 = COMMON/DEFTAB/ DEFRNG(10, 2), DEFCON(10, 2)
   = C
42 = COMMON/DEFOPS/DEFOPS(10)
   = C
43 = COMMON/DEFMOS/DEFMOS(16)
   = C
44 = SAVE /CPANEL/, /PIKBUF/, /SAVER/, /EPBBUF/, /SENDEL/
45 = SAVE /DEBUG/, /IOFVEC/, /CNTVEC/, /OPSVEC/
46 = SAVE /OPPARM/, /SENHEN/, /OPSPNT/, /CLOCK/, /WORK/, /DEFTAB/
47 = SAVE /PARAMS/, /ERRFLG/, /COMBUF/, /SENBUFF/, /CONTAB/, /MESTAB/, /DEFAULT/
48 = SAVE /SENINT/, /MESTB2/, /MESTB3/, /DEFOPS/
49 = SAVE /DEFMOS/
   = C
50 = EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
51 = EQUIVALENCE (OPSVEC(1), OPMODE), (OPSVEC(2), MODECT), (OPSVEC(3), PNGMSK)
52 = EQUIVALENCE (OPSVEC(4), OPSENR), (OPSVEC(5), RATECK)
53 = EQUIVALENCE (OPSVEC(6), STATUS), (TIME, TFLAG(4))
   = C
   C
54 LOGICAL*4 IOFCAS
   C
55 DIMENSION SENDEL(500)
   C

```

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

```

56      EQUIVALENCE (IOFVED(15), IOFCAS)
57      EQUIVALENCE (SENDEL(1), LODEL(1))
      C
      C *****
      C
58      STATUS=0
      C CLEAR DELAY POINTERS
59      DO 100 I=L 16
60      SENPTR(I)=0
61      100 CONTINUE
      C
      C CLEAR ERROR BUFFERS
62      DO 200 I=L 8
63      DO 200 J=L 10
64      ERRBUF(I,J)=0
65      200 CONTINUE
      C
      C CLEAR OPERATION POINTERS
66      DO 300 I=L 8
67      OPPTS(I)=DEFPNT(I)
68      300 CONTINUE
69      OPPTS(1)=1
      C
      C CLEAR CLOCK ARRAY
70      DO 400 I=L 6
71      TFLAG(I)=0
72      400 CONTINUE
      C
      C CLEAR PICK DATA
73      DO 500 I=L 10
74      PKFBUF(I)=0
75      PKRBUF(I)=0
76      500 CONTINUE
      C
      C CLEAR DELAY LINES
77      DO 600 I=L 500
78      SENDEL(I)=0
79      600 CONTINUE
      C
80      IF(IOFCAS) CALL TAPITZ
      C IF CASSETTE ON WRITE TAPE HEADER
      C
      C CMOS POINTER RESET, LSTPNT
81 /    DSCNTL(6)=0
      C
82 /    RETURN
83      END

```

MODULE INFORMATION:

CODE AREA SIZE	= 013AH	3140
VARIABLE AREA SIZE	= 0014H	200
MAXIMUM STACK SIZE	= 0002H	20
155 LINES READ		

FORTRAN COMPILER

ORIGINAL PAGE IS
OF POOR QUALITY

PAGE 1

0 PROGRAM ERROR(S) IN PROGRAM UNIT RESTRT

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

IS15-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT CNTFCD
 OBJECT MODULE PLACED IN FILEVSCNT.OBJ
 COMPILER INVOKED BY: FORT90 FILEVSCNT.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE CNTFCD
      C
      C CNTFCD IS THE FRONT DRUM CONTROL ALGORITHM
      C      THIS VERSION INCLUDES LCF, PICKS, HBS-CID, DEADEND, BIAS
      C
      C INPUTS: SENSOR DATA, CONTROL INFORMATION
      C OUTPUTS: FRONT DRUM CONTROL SIGNAL
      C CALLS: NOTHING
      C DESTROYS: NOTHING
      C MODIFIES: CONTROL OUTPUT FOR FRONT
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*4 CIB(80), KIB(8), DOB(6), TOB(128), PIB(2), POB(2), VOB(2)
4      INTEGER*4 PKFSAY, PKFBUF, PKRSAY, PKRBUF
      C
5      LOGICAL*4 IOFVCF
6      LOGICAL*4 CNMCID, CNMLCF, CNMPKF, CNMPKR, CNMPCF, CDUM(11), IOFVEC(38)
      C
7      COMMON/OPSPNT/OPPNTS(8)
      C
9      COMMON/CONBUF/ CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
9      COMMON/SAVER/LCX, FCNTL, FCDEL, TEMPOR(17)
      C
10     COMMON/SENMEM/ LSTPAS(3000), DSCNTL(16)
      C
11     COMMON/SENFNT/FDRPNT, FDRCNT, LCPNT, LCNT, PCPNT, PCNT, DUMMY(10)
      C
12     COMMON/PIKBUF/PKFBUF(10), PKRBUF(10), PKFSAY, PKRSAY
      C
13     COMMON/SENBUF/SIB(10), SDB(10)
      C
14     COMMON/PARAMS/DMAX, DMIN, ERLINT, CDSTEP, PIKAVE, FKSTEP, DEDED,
1      DEDED, ARMLNG, DRHRAD, LCPOS, PCPOS, CIDPOS, FTRLOC, DFDCT,
2      SMDSTP, DBIASF, DBIASR, ERKVAL, TRFCT
      C
15     COMMON/CNTVEC/CNMCID, CNMLCF, CNMPKF, CNMPKR, CNMPCF, CDUM
      C
16     COMMON/OPSVEC/ OPMODE, MODECT, RHMGR, OPSENR, RATECK, STAT,
1      FDVGN, FDCAL, ROYGN, RDCAL
      C
17     COMMON/SENDEL/LCDEL(40), PCDEL(60), FDRDEL(400)
      C
18     COMMON/IOFVEC/IOFVEC
      C
19     COMMON/ERRFLD/ERROR(8)
      C
20     SAVE .SENMEM/, /IOFVEC/
21     SAVE /OPSPNT/, /SENDEL/, /SAVER/, /PIKBUF/, /SENFNT/

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

22      SAVE /CNTVEC/, /OPSVEC/, /SENEUF/, /PPRMS/, /CONSEUF/
      C
23      EQUIVALENCE (SDB(1), DFD), (IOFVEC(10), IOFVCF)
24      EQUIVALENCE (SDB(6), CID), (SDB(2), LC), (SDB(4), FDRUM), (SDB(7), FEF)
25      EQUIVALENCE (DSCNTL(1), MAXSLT), (DSCNTL(2), STRTPT)
26      EQUIVALENCE (DSCNTL(3), STEPSZ), (DSCNTL(4), NEGMAX)
27      EQUIVALENCE (DSCNTL(5), POSMAX), (DSCNTL(6), LSTPNT)
28      EQUIVALENCE (DSCNTL(7), ISNIND), (DSCNTL(8), ISNPOS)
29      EQUIVALENCE (DSCNTL(9), CIDMAX), (DSCNTL(10), CIDMIN)
      C
      C*****
      C
      C SENSOR SELECT SECTION
      C
      C TEST IF LCF DATA USED
      C BEGIN BLOCK = SENSOR
30      IF((MODECT.EQ.1).OR.(MODECT.EQ.3)) THEN
      C
31      LCX=(LC*(20-TRGFCT+((10*FDRUM)/FDCAL)))/20
      C
      C BEGIN BLOCK = DELAY
32      IF(OPSENR.EQ.1) THEN
      C
      C OPSENR = 1 FOR DELAYED SENSORS
      C
      C DELAYED LCF ROUTINE
33      10  CONTINUE
      C
      C IF DFD >= OLD DFD + 1 OR MORE STEPS
34      IF((DFD-LCINT+0DFDCT).GT.DFDCT) THEN
35      LCDEL(MOD(LCPNT,LCPOS)+1)=LCX+((FDRUM+FDVGN)/10)*ARMLNG
36      LCPNT=LCPNT+1
37      LCINT=LCINT+1
38      GO TO 10
39      END IF
      C
      C TAKE LC FROM DELAYED BUFFER
40      LCX=LCDEL(MOD(LCPNT,LCPOS)+1)
      C
      C END SENSOR IF BLOCK
41      END IF
      C END BLOCK = DELAY
      C
      C *****
      C
      C ELSE NO LCF - RECORDED DATA MODE
42      ELSE
      C
43      IF((MODECT.EQ.2).OR.(MODECT.EQ.4)) THEN
      C LC DATA FROM CHOS BUFFER
44      INDEX=DFD+STRTPT
45      LCX=LSTPAS(INDEX)
      C IF NOT INITIALIZED
46      IF(LCX.EQ.(-32000)) THEN
47      LCX=0
      C ERROR 6 = NO LAST PASS DATA GOOD
48      ERROR(6)=1

```

```

49      END IF
      C
      C END STORED BLOCK
50      END IF
      C
51      END IF
      C END BLOCK = SENSOR
      C
      C *****
      C
      C BUFFER FRONT POSITION FOR SLAVE OF REAR (MODE 3)
52      20 CONTINUE
      C IF DFD >= OLD CFD + 1 OR MORE STEPS
53          IF((DFD-FDRCNT*(DFDCT).GT.DFDCT) THEN
54              FDRDEL(MOD(FDRPNT,FTRLOC)+1)=FORUM*ARMLNG
55              FDRPNT=FDRPNT+1
56              FDRCNT=FDRCNT+1
57              GO TO 20
58          END IF
      C
      C STORE CID/PC IN CMOS
      C
59      INDEX=CFD+STRPT-INSPOS
60      IF((INDEX.LE.POSMAX).AND.(INDEX.GE.NEGMAX)) THEN
      C OK IN RANGE
61          IF(INDEX.NE.LSTPNT) THEN
62              DO 50 I=LSTPNT+1,INDEX
63                  LSTPAS(I)=SDB(INSIND)
64          50 CONTINUE
65              LSTPNT=INDEX
66          END IF
67          ELSE
68              ERROR(6)=10
69          END IF
      C
      C END SENSOR SEGMENT
      C
      C*****
      C
      C CONTROL ALGORITHM SECTION
      C
      C IF AUTO MODE DO CONTROL
      C BEGIN BLOCK = CONTROL
70      IF(OPMODE.EQ.1) THEN
      C
71          FCNTL=0
72          VOB(1)=0
      C
      C *****
      C
      C TEST IF PICK CONTROL
      C BEGIN BLOCK = PICKS
73      IF(CNMPKF) THEN
      C
      C TEST IF AVERAGE PICKS
74      IF(PIKAVE.GT.0) THEN
      C STORE PICK DATA FOR AVERAGING

```

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

```

75      I=MOD(DEFPTS(5),PIKAVE)
76      PKFEUF(I+1)=PKF+10
77      J=0
      C AVERAGE PICK READINGS
78      DO 100 I=1,PIKAVE
79      J=J+PKFEUF(I)
80      100 CONTINUE
81      J=J/PIKAVE
      C
      C ELSE NO AVERAGING ON PICK DATA
82      ELSE
83      J=PKF+10
84      END IF
      C
      C TEST IF OVER OR UNDER THRESHOLDS FOR CHANGES
85      IF(J.GE.7) THEN
86      J=1
87      GO TO 200
88      END IF
      C
      C TEST IF COAL
89      IF(J.LE.3) THEN
90      J=-1
91      GO TO 200
92      END IF
      C
      C ELSE KEEP OLD VALUE
93      J=PKFSAY
      C
94      200 PKFSAY=J
      C
      C SHIFT LC DUE TO PICKS
95      LCX=LCX+PKSTEP*J
      C
96      END IF
      C END BLOCK = PICKS
      C
      C *****
      C
      C BEGIN BLOCK = CID
97      IF(CNMCID) THEN
      C
      C DO CID CONTROL AS PICKS ARE DONE
      C
      C IF CID .GT. SET POINT MOVE DRUM DOWN
98      IF(CID.GT.CIDMAX) LCX=LCX-CDSTEP
      C
      C IF CID .LT. MIN SET POINT MOVE DRUM UP
99      IF(CID.LT.CIDMIN) LCX=LCX+CDSTEP
      C
100     END IF
      C END BLOCK = CID
      C
      C *****
      C
      C BEGIN BLOCK = LCF
101     IF(CNMLCF) THEN

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C
      C TEST IF NULLING CONTROL
102      IF(NODECT.EQ.9) FCNTL=LCK
      C
      C TEST IF NORMAL REAL TIME CONTROL
103      IF((NODECT.EQ.1).OR.(NODECT.EQ.3)) FCNTL=LCK-FERUM*APPLNG
      C
104      ELSE
      C NO LCF DATA AVAILABLE DIRECTLY
      C
      C TEST IF STORED DATA (IN CMOS)
105      IF((NODECT.EQ.4).OR.(NODECT.EQ.2)) FCNTL=LCK-FERUM*APPLNG
      C
      C END LCF BLOCK
106      END IF
      C
      C *****
      C
      C CONTROL SIGNAL GENERATION SECTION
      C
      C TEST IF OUTPUT ENABLED
      C BEGIN BLOCK = OUTPUT
107      IF(10FYCF) THEN
      C
      C OUTPUT BIAS AND DEAD BAND
108      FCDELT=(FCNTL-OBIA5F)
      C
      C TEST IF DEAD BAND ON OUTPUT
109      IF(DEDBOF.GT.0) *SET
110      IF(IABS(FCDELT).LT.DEDEBF) THEN
      C IF CONTROL .LT. DEAD BAND SET TO 0
111      FCNTL=0
112      GO TO 500
113      END IF
114      END IF
      C
      C *****
      C
      C GENERATE VALVE CONTROL
115      IF(FCDELT.GT.0) THEN
116      VOB(1)=1
      C DRUM LOW , GO UP
117      ELSE
118      VOB(1)=-1
      C DRUM HIGH , GO DOWN
119      END IF
      C
120      500 CONTINUE
      C
121      END IF
      C END BLOCK = OUTPUT
      C
122      END IF
      C END BLOCK = CONTROL
      C
123      RETURN
124      END

```

MODULE INFORMATION

ORIGINAL PAGE IS
OF POOR QUALITY

CODE AREA SIZE = 010FH 8910
VARIABLE AREA SIZE = 000AH 100
MAXIMUM STACK SIZE = 0004H 40
279 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT CNTRFD

ORIGINAL PAGE IS
OF POOR QUALITY

TS1E-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT CNTFRD
OBJECT MODULE PLACED IN :F1:VCSCHT.OBJ
COMPILER INVOKED BY: FORT90 :F1:VCSCHT.FOR

```

1      SUBROUTINE CNTFRD
      C
      C CNTFRD COMPUTES THE REAR DRUM CONTROL SIGNAL. ALGORITHM IS
      C   CONTROLLED BY INPUTS AND MODE VECTOR
      C
      C INPUTS: SENSOR DATA, CONTROL INFORMATION
      C OUTPUTS: REAR DRUM CONTROL SIGNAL
      C CALLS: NOTHING
      C DESTROYS: NOTHING
      C MODIFIES: CONTROL OUTPUT FOR REAR
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
4      LOGICAL*1 IOFVEC(20), IOFYCR
5      LOGICAL*1 CNMCID, CNMLCF, CNMPKF, CNMPKR, CNMPCF, CNMYCF, CNMYCP, CDUM(9)
      C
6      COMMON/COMBUF/CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
7      COMMON/SAVER/PCX, RCNTL, RCOELT, TEMPOR(17)
      C
8      COMMON/SENMEM/ LSTPAS(3000), OSCNTL(16)
      C
9      COMMON/SENENT/FORENT, FORCNT, LCPNT, LCCHT, PCPNT, PCCNT, PDUM(10)
      C
10     COMMON/SENBUF/SID(10), SDB(10)
      C
11     COMMON/PARAMS/DMAX, DMIN, ERLINT, SEAR, PIKAVE, PKSTEP,
12     1 DEDECF, DEDECR, ARMLNG,
13     2 DMRAD, LCPQS, PCQS, CIDQS, FTRLOC, DFDCT, SINDSTP,
14     3 DBIASF, DBIASR, SRKVAL, TRGFCT
      C
15     COMMON/CNTVEC/CNMCID, CNMLCF, CNMPKF, CNMPKR, CNMPCF, CNMYCF, CNMYCP, CDUM
      C
16     COMMON/OPSVEC/OPMODE, MODECT, RNQMSK, OPSENR, RATECK, STAT,
17     1 FVQSH, FDCAL, RQVGN, RDCAL
      C
18     COMMON/SENDEL/LCDEL(40), PCDEL(60), FORDEL(400)
      C
19     COMMON/IOFVEC/IOFVEC
      C
20     SAVE /OPSPNT/, /IOFVEC/
      C
21     SAVE /SENDEL/, /SAVER/, /SENMEM/, /PIKBUF/, /SENENT/
22     SAVE /CNTVEC/, /OPSVEC/, /SENBUF/, /PARAMS/, /COMBUF/
      C
23     EQUIVALENCE (SDB(1), OFD), (IOFVEC(11), IOFYCR)
24     EQUIVALENCE (SDB(2), PC), (SDB(8), PKR), (SDB(5), RDRUM)
      C
      C*****

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C
      C SENSOR SELECT SECTION
      C
      C TEST IF PCF DATA USED
      C BEGIN BLOCK = SENSOR
21      IF(MODECT.LE.1) THEN
      C
22      PCX=(PC*(20-TRGCT*((10*RDUM)/FDCAL)))/20
      C
      C TEST IF DELAYED SENSOR
      C
      C BEGIN BLOCK = DELAY
23      IF(OPSENR.EQ.1) THEN
      C
      C PCF DELAY ROUTINE
24      10 CONTINUE
      C IF DFD )= OLD DFD + 1 (OR MORE STEPS
      IF((DFD-PCNT*(F*ST).GT.DFDFCT) THEN
25      PCDEL(MOD(PCNT,PCPOS)+1)=PCX+((RDUM*RDVGN)/10)*ARMING
26      PCNT=PCNT+1
27      PCNT=PCNT+1
28      GO TO 10
29      END IF
      C
      C TAKE PC DATA FROM DELAY BUFFER
31      PCX=PCDEL(MOD(PCNT,PCPOS)+1)
      C
      C END BLOCK = DELAY
32      END IF
      C
      C *****
      C
      C ELSE NO PCF - RECORDED DATA MODE
23      ELSE
      C
      C TEST IF STORED DATA
      C IF((MODECT.EQ.2).OR.(MODECT.EQ.4)) PCX=PC
      C
      C TEST IF SLAVED DRUM
34      IF(MODECT.EQ.3) PCX=FDRDEL(MOD(FDRPNT,FTRLOC)+1)
      C
      C END BLOCK = SENSOR
35      END IF
      C
      C END SENSOR SEGMENT
      C
      C *****
      C
      C CONTROL ALGORITHM SECTION
      C
      C IF AUTO DO CONTROL
      C BEGIN BLOCK = CONTROL
36      IF(OPMODE.EQ.1) THEN
      C
37      RCNTL=0
38      VOB(2)=0
      C

```



```

      C TEST IF REAR PCF CONTROL
      C BEGIN BLOCK = PCF
39      IF((CHMPCF) THEN
      C
      C TEST IF NULLING CONTROL
40      IF((MODECT.EQ.0) RCNTL=PCX
      C
      C TEST IF NORMAL REAL TIME CONTROL
41      IF((MODECT.EQ.1).OR.(MODECT.EQ.3)) RCNTL=PCX-ADRUM*ARMING
      C
      C *****
      C
42      ELSE
      C NO PCF DATA AVAILABLE
      C
      C TEST IF SLAVED DRUM CONTROL
43      IF((MODECT.EQ.3) RCNTL=PCX-ADRUM*ARMING
      C
      C TEST IF STORED DATA (CMOS)
      C REAR IS SAME AS 1,3 FOR NOW
44      IF((MODECT.EQ.4).OR.(MODECT.EQ.2)) RCNTL=PCX-ADRUM*ARMING
      C
      C END BLOCK = PCF
45      END IF
      C
      C *****
      C
      C CONTROL SIGNAL GENERATION SECTION
      C
      C TEST IF OUTPUT ENABLED FOR VALVE
      C BEGIN BLOCK = OUTPUT
46      IF((IOFVCR) THEN
      C
      C OUTPUT BIAS AND DEAD BAND
47      RDELT=(RCNTL-GBIASR)
      C
      C DEAD BAND ON CONTROL OUTPUT
48      IF((DEDBDR.GT.0) THEN
49      IF((IABS(RDELT).LT.DEDBDR) THEN
50      RCNTL=0
51      GO TO 500
52      END IF
53      END IF
      C
      C *****
      C
      C GENERATE VALVE CONTROL
54      IF((RDELT.GT.0) THEN
      C DRUM LOW , GO UP
55      VOB(2)=1
56      ELSE
      C DRUM HIGH , GO DOWN
57      VOB(2)=-1
58      END IF
      C
59      500 CONTINUE
      C END SIGNAL BLOCK

```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      C END BLOCK = OUTPUT  
60      END IF  
      C  
      C END CONTROL BLOCK  
      C END BLOCK = CONTROL  
61      END IF  
      C  
62      RETURN  
63      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

MODULE INFORMATION:

```
CODE AREA SIZE    = 01F3H    4990  
VARIABLE AREA SIZE = 0000H    00  
MAXIMUM STACK SIZE = 0004H    40  
174 LINES READ
```

0 PROGRAM ERROR(S) IN PROGRAM UNIT CNTRPD

ISIS-11 FORT80N-80 V2.1 COMPILATION OF PROGRAM UNIT SENRLM
 OBJECT MODULE PLACED IN :F1:VCSOINT.OBJ
 COMPILER INVOKED BY: FORT80 :F1:VCSOINT.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE SENRLM
      C
      C SENRLM IS THE SENSOR INPUT RATE LIMIT ERROR DETECTOR ROUTINE
      C
      C INPUTS: SDB
      C OUTPUTS: ERROR STATUS
      C CALLS: NOTHING
      C DESTROYS: NOTHING
      C MODIFIES: SIB, SDB, ERROR FLAG
      C VERSION: 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      COMMON/SENBUF/SIB(10), SDB(10)
      C
4      COMMON/OPSVEC/OPMODE, MODECT, RNHNSK, OPSENP, RATECK, STATUS, OOLM(4)
      C
5      COMMON/ERRFLG/ ERROR(8)
      C
6      COMMON/PARAMS/PDUM(10), SRKVAL, DUND
      C
7      SAVE /PARAMS/
8      SAVE /OPSVEC/, /ERRFLG/, /SENBUF/
      C
9      EQUIVALENCE (SDB(3), LCF), (SDB(2), PCF)
      C
      C*****
      C
      C IF OLD START INITIALIZE OLD VALUES
      C
10     IF(STATUS.EQ.0) THEN
11         LCOLD=LCF
12         PCOLD=PCF
13         STATUS=STATUS+1
14     END IF
      C
      C IF IN AUTOMATIC DO ERROR TEST
      C
15     IF(OPMODE.EQ.1) THEN
      C TEST IF RATE TOO LARGE ON INPUTS
      C ERROR 4 = INPUT SENSOR RATE OF CHANGE TOO HIGH
16         IF(ABS(LCF-LCOLD).GE.SRKVAL) ERROR(4)=1
      C
17         IF(ABS(PCF-PCOLD).GE.SRKVAL) ERROR(4)=ERROR(4)+10
      C
18     END IF
      C
19     IF(ERROR(4).NE.0) RETURN
      C
      C USE OLD VALUES IF ERROR
20     PCOLD=PCF
  
```

```
21      LOCLD=LCF  
      C  
22      RETURN  
23      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

MODULE INFORMATION:

```
CODE AREA SIZE   = 007FH   127D  
VARIABLE AREA SIZE = 0004H   4D  
MAXIMUM STACK SIZE = 0002H   2D  
55 LINES READ
```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SENRLM

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ORIGINAL PAGE IS
OF POOR QUALITY

TEIS-II FORTAN-90 V2.1 COMPILATION OF PROGRAM UNIT DEUGCI
OBJECT MODULE PLACED IN :F1:VCSDBG.OBJ
COMPILER INVOKED BY: FORT80 :F1:VCSDBG.FOR

```

1      SUBROUTINE DEUGCI
      C
      C DEUGCI READS DEBUG CONTROL INFORMATION FROM CONTROL/DISPLAY PANEL
      C
      C INPUTS: DEBUG CONTROL VECTOR
      C OUTPUTS: NONE
      C CALLS: INPCDU, OUTDSP, PAUSE
      C DESTROYS: NOTHING
      C MODIFIES: DEBUG CONTROL VECTOR
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      CHARACTER*1 DSFCOD(10)
4      CHARACTER*4 DOBC, DOBN
5      CHARACTER*4 ERRMES(20), DBGMES(20), RUNMES(20)
      C
6      INTEGER*1 BITS(8)
7      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), YOB(2)
      C
8      LOGICAL*1 NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALERT
      C
9      COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALERT
      C
10     COMMON/OPARM/MDUM(18), NMKDBG, MDUM(18)
      C
11     COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, POB, YOB
      C
12     COMMON/MESTAB/ ERRMES, DBGMES, RUNMES, DSFCOD
      C
13     COMMON/WORK/ TENPIO, DUMMY(19)
      C
14     SAVE /COMBUF/, /MESTAB/, /WORK/, /CPANEL/, /OPARM/
      C
15     EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
      C
16     EQUIVALENCE (TENPIO, BITS(1))
      C
      C*****
      C
      C LOAD DOB WITH PROMPT
      C
      C *DEBUG!
17     DOBC=RUNMES(7)
18     DOBN=RUNMES(8)
      C
      C DISPLAY MESSAGE
      C
19     CALL OUTDSP
      C
20     CALL PAUSE

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C
      C PROMPT FOR NMYDBG DEEUG OPERATIONS
      C
      C LOAD DOBC WITH DEBUG CODE
      C
      C 0DEGXXX
21      DOBC=DBGMS(1)
      C
      C M) FOR ALL DEBUG TYPES
22      DO 100 I=L,NMYDBG
      C
      C LOAD DOBN WITH SECOND HALF OF PROMPT
23      DOBN=DBGMS(I+1)
      C
      C DISPLAY AND WAIT FOR RESPONSE
      C
24      CALL OUTDSP
25      CALL INPCDU
      C
      C TEST IF ADJ PRESSED
26      IF(ADJUST) THEN
27          TEMP10=1
28          GO TO 1000
29      END IF
      C
      C IF RST ; EXIT
30      IF(RESET) THEN
31          TEMP10=0
32          GO TO 1000
33      END IF
      C
      C ELSE IF NKT ; CONTINUE
34      100 CONTINUE
      C
      C DEFAULT EXIT
35      TEMP10=0
      C
36      1000 CONTINUE
37      RETURN
38      END

```

MODULE INFORMATION:

```

CODE AREA SIZE   = 00A5H   163D
VARIABLE AREA SIZE = 0004H    4D
MAXIMUM STACK SIZE = 000AH   10D
91 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT DEUGCI

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-11 FORT80N-80 V2.1 COMPILATION OF PROGRAM UNIT DEUGCO
OBJECT MODULE PLACED IN :F1:VCSDBG.OBJ
COMPILER INVOKED BY: FORT80 :F1:VCSDBG.FOR

```

1      SUBROUTINE DEUGCO
      C
      C DEUGCO OUTPUTS DEBUG INFORMATION TO CONTROL/DISPLAY PANEL
      C
      C INPUTS:  DEBUG CONTROL VECTOR
      C OUTPUTS: DEBUG MESSAGES
      C CALLS:  OUTDSP,CLOCK
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      CHARACTER*1 DSPCCO(10)
4      CHARACTER*4 DOBC,DOBN
5      CHARACTER*4 ERRMES(20),DEGMES(20),RUNMES(20)
      C
6      INTEGER*4 CIB(80),KIB(8),DOB(8),TOB(128),PIB(2),POB(2),VOB(2)
      C
7      COMMON/COMBUF/ CIB,KIB,DOB,TOB,PIB,POB,VOB
      C
8      COMMON/NESTAB/ ERRMES,DEGMES,RUNMES,DSPCCO
      C
9      SAVE /COMBUF/,/NESTAB/
      C
10     EQUIVALENCE (DOB(1),DOBC),(DOBN,DOB(5))
      C
      C*****
      C
      C LOAD DOB WITH PROMPT
      C
11     DOBC=DEGMES(1)
12     DOBN=RUNMES(6)
      C @DBG-OK*
      C
      C DISPLAY MESSAGE
      C
13     CALL OUTDSP
      C
      C USE TIME OUT FEATURE
14     CALL CLOCK
      C
15     RETURN
16     END

```

MODULE INFORMATION:

CODE AREA SIZE = 0020H 45D
VARIABLE AREA SIZE = 0000H 0D

FORTRAN COMPILER

ORIGINAL PAGE IS
OF POOR QUALITY

MAXIMUM STACK SIZE = 000AH 100
44 LINES READ

0 PROGRAM ERRORS IN PROGRAM UNIT DEUGCO

ORIGINAL PAGE IS
OF POOR QUALITY.

ISIS-II FORTAN-90 V2.1 COMPILATION OF PROGRAM UNIT DEUGOT
OBJECT MODULE PLACED IN :F1:VCSDBG.OBJ
COMPILER INVOKED BY: FORT90 :F1:VCSDBG.FOR

```

1      SUBROUTINE DEUGOT
      C
      C DEUGOT OUTPUTS DEBUG CONTROL INFORMATION TO FRONT AND REAR DRUMS
      C
      C INPUTS: DEBUG CONTROL INFORMATION
      C OUTPUTS: DRUM CONTROL SIGNALS
      C CALLS: OUTDSP, INPCDU, CNTOUT
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
4      CHARACTER*1 DSPCOD(10), DOB4
5      CHARACTER*4 DOBC, DOBN
6      CHARACTER*4 ERRMES(20), DBGMES(20), RUNMES(20)
      C
7      LOGICAL*1 NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALERT
8      LOGICAL*1 IOFVEC, IOFVOT
9      LOGICAL*1 CNTVEC(16), CNMFRT, CNMFER, CNMOUT, SAVVEC(40)
      C
10     COMMON/IOFVEC/IOFVEC(20)
      C
11     COMMON/NESTAB/ERRMES, DBGMES, RUNMES, DSPCOD
      C
12     COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALERT
      C
13     COMMON/COMBUF/CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
14     COMMON/OPPARM/NDUN(17), NMXCNM, MDUN(11)
      C
15     COMMON/CNTVEC/CNTVEC
      C
16     COMMON/SAVER/SAVVEC
      C
17     SAVE /IOFVEC/, /SAVER/
18     SAVE /NESTAB/, /COMBUF/, /CNTVEC/, /OPPARM/, /CPANEL/
      C
19     EQUIVALENCE (IOFVOT, IOFVEC(12))
20     EQUIVALENCE (CNTVEC(8), CNMFRT), (CNTVEC(9), CNMFER), (CNTVEC(10), CNMOUT)
21     EQUIVALENCE (DOB(1), DOBC), (DOBN, DOB(5)), (DOB(4), DOB4)
      C
      C*****
      C
      C SAVE CONTROL MASK VECTOR AND FORCE ELEMENTS
      C
22     DO 100 I=1, NMXCNM
23     SAVVEC(I)=CNTVEC(I)
24     CNTVEC(I)= .TRUE.

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

25 100 CONTINUE
    C
26     SAVVEC(NH(NCM)+1)=IOFVOT
27     IOFVOT= TRUE
    C
    C LOAD DISPLAY WITH PROMPT
    C
28 150 CONTINUE
29     DOBC=DEGMES(2)
30     DOB4=DEPCOD(9)
31     DOBN=DEPMES(12)
    C OUT=NO!
    C
    C DISPLAY MESSAGE
    C
32     CALL OUTDSP
    C
    C WAIT FOR OPERATOR RESPONSE
    C
    C ADJUST = UP
    C NEXT = DOWN
    C RESET = STOP
    C
33     CALL INPCDU
34     IF(ADJUST) THEN
35         VOB(1)=1
36         VOB(2)=1
    C UP
37         DOBN=DEGMES(15)
38     ELSE
39         IF(NEXT) THEN
40             VOB(1)=-1
41             VOB(2)=-1
    C DOWN
42         DOBN=DEGMES(16)
43     ELSE
44         VOB(1)=0
45         VOB(2)=0
46         DOBN=DEGMES(11)
    C END
47     END IF
48     END IF
    C
    C
    C OUTPUT VALVE CONTROL SIGNALS
    C
49     CALL CNTOUT
    C
    C WAIT FOR OPERATOR RESPONSE
    C
50     CALL OUTDSP
    C
    C OUT?END? OUT? UP OUT?DOWN
51     CALL INPCDU
    C
    C RESET VOB TO ZERO
    C
52     VOB(1)=0

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

53      VOB(2)=0
      C
      C OUTPUT VOB SIGNAL
      C
54      CALL CNTOUT
      C
55      IF (ADJUST) GO TO 150
      C
      C RESTORE CONTROL MASK VECTOR
      C
56      DO 200 I=L,NMACHM
57      CNTVEC(I)=SAVVEC(I)
58      200 CONTINUE
59      IOFVOT=SAVVEC(NMACHM+1)
      C
60      RETURN
61      END

```

MODULE INFORMATION:

```

CODE AREA SIZE   = 0156H   342D
VARIABLE AREA SIZE = 0004H    4D
MAXIMUM STACK SIZE = 000AH   10D
125 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT DEUGDT

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT DEUGIN
 OBJECT MODULE PLACED IN :F1:VCSE08G.OBJ
 COMPILER INVOKED BY: FORT90 :F1:VCSE08G.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE DEUGIN (FLAG)
      C
      C DEUGIN READS SENSOR DATA VIA I/O ROUTINES AS PER INPUT
      C
      C INPUTS: SENSOR BUFFER, CONTROL VECTOR
      C OUTPUTS: NONE
      C CALLS: SENINP, PAUSE, OUTDSP, FORMAT, CONVRT, INPCDU
      C DESTROYS: NOTHING
      C MODIFIES: SENSOR INPUT DATA SIB, SDB
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*4 CBYTE
4      INTEGER*4 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
5      CHARACTER*1 DESPCOD(10), DOB4
6      CHARACTER*4 DOBC, DOBB
7      CHARACTER*4 ERRHES(20), DEGHES(20), RUNMES(20)
      C
8      LOGICAL*1 NEXT, AUTO, RUN, RESET, ADJUST, MANUAL, PHRFAL, CALERT
9      LOGICAL*1 IOFVEC(20), SAYVEC(40)
      C
10     COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALERT
      C
11     COMMON/SENBUF/SIB(10), SDB(10)
      C
12     COMMON/COMBUF/CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
13     COMMON/IOFVEC/IOFVEC
      C
14     COMMON/OPARM/LDUM(10), NMKSEN, NDUM(5), NMXIOW, NDUM(12)
      C
15     COMMON/WORK/TEMP10, NDUM(3), ZDUM(16)
      C
16     COMMON/SAVER/SAYVEC
      C
17     COMMON/NESTAB/ERRHES, DEGHES, RUNMES, DESPCOD
      C
18     SAVE /CPANEL/, /SAVER/, /OPARM/
19     SAVE /COMBUF/, /IOFVEC/, /NESTAB/, /WORK/, /SENBUF/
      C
20     EQUIVALENCE (DOB(5), DOBB), (DOB(4), DOB4)
21     EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
      C
      C*****
      C
      C LOAD DOB WITH PROMPT FOR DISPLAY
      C
      C FLAG = 1 FOR DATA CONVERSION
22     IF(FLAG.NE.0) THEN

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

22      DOB0=DEGMES(4)
24      ELSE
25      DOB0=DEGMES(3)
26      END IF
27      DOB8=RUAMES(20)
29      DOB4=DEPCOD(9)
      C IMP= ##!
      C DISPLAY MESSAGE
      C
29      CALL OUTDSP
      C
      C WAIT FOR OPERATOR RESPONSE
      C
30      CALL PAUSE
      C
      C SAVE I/O MASK VECTOR AND SET TRUE
      C
31      DO 100 I=L,NMXIOF
32      SAVVEC(I)=IOFVEC(I)
33      IOFVEC(I)=.TRUE
34      100 CONTINUE
      C
      C CALL SENSOR INPUT ROUTINE
      C
35      150 CALL SENIMP
      C
36      IF(FLAG.NE.0) CALL CONVRT
      C
      C DISPLAY VALUE READ IN FOR EACH I/O CHANNEL
      C
37      DO 200 I=L,NMXSEN
38      IF(FLAG.EQ.0) THEN
39      DOBN=SIB(I)
40      ELSE
41      DOBN=SDB(I)
42      END IF
43      DOB(3)=I+430H
      C INX=ZZZZ
      C
      C FORMAT HEX VALUE INTO ASCII IN DOB
      C
44      CALL FORMAT(DOBN)
      C
45      CALL OUTDSP
      C
      C WAIT FOR OPERATOR RESPONSE
      C
46      CALL INPCDU
47      IF(RESET) GO TO 250
48      IF(ADJUST) GO TO 150
      C
49      200 CONTINUE
      C
      C RESTORE I/O MASK VECTOR
      C
50      250 DO 300 I=L,NMXIOF

```

FORTRAN COMPILER

```
51      IOFVEC(I)=SAWVEC(I)
52      200  CONTINUE
        C
53      RETURN
54      END
```

ORIGINAL PAGE IS
OF POOR QUALITY

MODULE INFORMATION:

```
CODE AREA SIZE   = 0162H   354D
VARIABLE AREA SIZE = 0007H    7D
MAXIMUM STACK SIZE = 000AH   10D
113 LINES READ
```

0 PROGRAM ERROR(S) IN PROGRAM UNIT DEUGIN

ISIS-II FORTPAN-80 V2.1 COMPILATION OF PROGRAM UNIT DEBUGCK
 OBJECT MODULE PLACED IN :F1:VCSDBG.OBJ
 COMPILER INVOKED BY: FORT80 :F1:VCSDBG.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE DEBUGCK
      C
      C DEBUGCK CHECKS THE CLOCK TIMER FOR VCS SYNCHRONIZATION
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: CLOCK, INTREN
      C DESTROYS: NOTHING
      C MODIFIES: DOB
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 TFLAG(6)
4      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
5      CHARACTER*4 DOBC, DOBN, ERRMES(20), DEGMES(20), RUNMES(20)
6      CHARACTER*1 DSPCOD(10)
      C
7      COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
8      COMMON/CLOCK/ TFLAG
      C
9      COMMON/MESTAB/ ERRMES, DEGMES, RUNMES, DSPCOD
      C
10     SAVE /COMBUF/, /MESTAB/, /CLOCK/
      C
11     EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
      C
      C*****
      C
      C ENABLE CLOCK INTERRUPTS
      C
12     CALL INTREN
13     ISAV=IOFCAS
14     IOFCAS=0
15     DOBC=DEGMES(8)
16     DOBN=DEGMES(7)
      C CLK?I/O?
      C
17     CALL CLOCK
      C
18     IOFCAS=ISAV
      C
19     RETURN
20     END
  
```

MODULE INFORMATION:

FORTRAN COMPILER

ORIGINAL PAGE IS
OF POOR QUALITY

CODE AREA SIZE = 003FH 63D
VARIABLE AREA SIZE = 0004H 4D
MAXIMUM STACK SIZE = 0000H 10D
45 LINES READ

0 PROGRAM ERROR: IN PROGRAM UNIT DEUSCH

ISIS-II F0PTAN-90 V2.1 COMPILATION OF PROGRAM UNIT DEUGIO
 OBJECT MODULE PLACED IN F1:VCSDBG.OBJ
 COMPILER INVOKED BY: FORT90 F1:VCSDBG.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE DEUGIO
      C
      C DEUGIO READS AN I/O CHANNEL OR PORT AND DISPLAYS VALUE
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: OUTDSP, PAUSE, INPCIU, SENINP, FORMAT
      C DESTROYS: NOTHING
      C MODIFIES: DOB, TEMPIO
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 BITS(8)
4      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), FOB(2), VOB(2)
5      INTEGER*1 DSPCOD(10)
      C
6      LOGICAL*1 NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALBRT
      C
7      CHARACTER*4 ERRMES(20), DEGMES(20), RUNMES(20), DOBC, DOBN
      C
8      COMMON/SENBUF/SIB(10), SDB(10)
      C
9      COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PHRFAL, RUN, CALBRT
      C
10     COMMON/WORK/ BITS, CUM(16)
      C
11     COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, FOB, VOB
      C
12     COMMON/NESTAB/ERRMES, DEGMES, RUNMES, DSPCOD
      C
13     SAVE /WORK/
14     SAVE /COMBUF/, /NESTAB/, /CPANEL/, /SENBUF/
      C
15     EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
16     EQUIVALENCE (DOB(8), DOB(5))
      C
      C*****
      C
      C LOAD DOB WITH PROMPT
      C
17     DOBC=RUNMES(1)
18     DOBN=DEGMES(14)
      C ?I/OCHA!
      C
19     CALL OUTDSP
      C
20     CALL PAUSE
      C
21     NMXIOC=8
      C

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

22      DO 100 I=L,NMX100
      C
      C LOAD DOB WITH PROMPTS
      C
22      DOBC=ERRMES(I+1)
24      DOB(1)=DEFDOB(6)
25      DOBN=RUNMES(20)
      C GXXX ##
      C
26      CALL OUTDSP
      C
27      CALL INPCDU
      C
      C TEST IF UPDATE
      C
      C IF RST EXIT
28      IF(RESET) GO TO 1000
      C TEST IF ADJ PRESSED
29      IF(ADJUST) THEN
      C
30      50  CALL SENINP
      C
31      TEMPIO=SIB(1)
      C
32      75  DOBX=TEMPIO
      C
      C FORMAT VALUE AND DISPLAY
33      CALL FORMAT(DOBX)
      C
34      CALL OUTDSP
      C
      C WAIT FOR OPERATOR
35      CALL INPCDU
      C
      C TEST IF NXT PRESSED CONTINUE LOOP
36      IF(NEXT) GO TO 100
      C
      C TEST IF RESET , EXIT
37      IF(RESET) GO TO 1000
      C ELSE ADJUST , READ AGAIN
38      GO TO 50
39      END IF
      C
40      100  CONTINUE
      C
      C ELSE NEXT, CONTINUE
41      1000 CONTINUE
42      RETURN
43      END

```

MODULE INFORMATION:

CODE AREA SIZE	= 0007H	215D
VARIABLE AREA SIZE	= 0006H	20
MAXIMUM STACK SIZE	= 000AH	100

FORTRAN COMPILER

100 LINES READ

0 PROGRAM ERRORS IN PROGRAM UNIT DEBUG

ORIGINAL PAGE IS
OF POOR QUALITY

PAGE 15

ISIS-II FORTRAN-80 V2.1 COMPILATION OF PROGRAM UNIT DEUGTP
 OBJECT MODULE PLACED IN :F1:VCSDBG.OBJ
 COMPILER INVOKED BY: FORT80 :F1:VCSDBG.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE DEUGTP
      C
      C DEUGTP CHECKS OPERATION OF THE TAPE CASSETTE
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: ATOCRS, OUTDSP, MSEC2H, PAUSE
      C DESTROYS: NOTHING
      C MODIFIES: DOB, TOB
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
4      INTEGER*1 CNTL(4)
      C
5      CHARACTER*1 DSPCOD(10)
6      CHARACTER*4 DOBC, DOBN, ERRMES(20), DGMES(20), RUNMES(20)
      C
7      COMMON/SENBUF/SIB(10), SDB(10)
      C
8      COMMON/COMBUF/CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
9      COMMON/MESTAB/ ERRMES, DGMES, RUNMES, DSPCOD
      C
10     COMMON/WORK/ TEMP10, DUMMY(19)
      C
11     SAVE /MESTAB/, /COMBUF/, /WORK/
12     SAVE /SENBUF/
      C
13     EQUIVALENCE (DUMMY(1), CNTL(1))
14     EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
      C
      C*****
      C
15     CNTL(2)=3
16     CNTL(1)=2
17     TEMP10=0
18     CALL ATOCRS(TEMP10)
19     CALL MSEC2H
20     IF (CNTL(2).NE.0) THEN
21       DOBC=ERRMES(15)
22       DOBN=RUNMES(15)
23       CALL OUTDSP
24       CALL PAUSE
25     END IF
26     RETURN
27     END
  
```

FORTPAN COMPILER

PAGE 17

MODULE INFORMATION

ORIGINAL PAGE IS
OF POOR QUALITY

CODE AREA SIZE = 0055H 850
VARIABLE AREA SIZE = 0000H 00
MAXIMUM STACK SIZE = 0000H 100
48 LINES REMO

0 PROGRAM ERROR(S) IN PROGRAM UNIT DEUGTP

TS15-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT DEBUNE
 OBJECT MODULE PLACED IN :F1.VCSD80.OBJ
 COMPILER INVOKED BY: FORT90 :F1.VCSD80.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE DEBUNE
      C
      C DEBUNE DISPLAYS MEMORY TO THE OPERATOR VIA THE DISPLAY
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: OUTDSP, FORMAT, INPCDU, AGETIT, APUTIT, INPHEX
      C DESTROYS: NOTHING
      C MODIFIES: DOB, MEMORY
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*4 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), FOB(2), VOB(2)
4      INTEGER*4 BITS(8)
      C
5      LOGICAL*4 NEXT, ADJUST, RESET, AUTO, MANUAL, PWRFAL, FUN, CALERT
      C
6      CHARACTER*1 DSPC00(10), DOB4
7      CHARACTER*4 DOBC, DOBN, ERRMES(20), DEGMES(20), RUNMES(20)
      C
8      COMMON/CPANEL/ NEXT, ADJUST, RESET, AUTO, MANUAL, PWRFAL, FUN, CALERT
      C
9      COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, FOB, VOB
      C
10     COMMON/WORK/ TEMP10, DUMMY(19)
      C
11     COMMON/NESTAB/ ERRMES, DEGMES, RUNMES, DSPC00
      C
12     SAVE /COMBUF/, /NESTAB/, /CPANEL/, /WORK/
      C
13     EQUIVALENCE (TEMP10, BITS(1))
14     EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
15     EQUIVALENCE (DOB(5), DOBX), (DOB4, DOB(4))
16     EQUIVALENCE (DOB(2), DOBZ)
      C
      C*****
      C
      C LOAD DOB WITH MEMORY DUMP MESSAGE
      C
17     DOBC=DEGMES(9)
18     DOBN=DEGMES(10)
      C MEN?DUMP
      C
19     CALL OUTDSP
      C
20     CALL INPCDU
      C
      C LOAD DOB WITH MEMORY ADDRESS REQUEST
      C
21     100 DOBC=DEGMES(20)

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

22      DOBN=RUNMES(20)
      C ADP? ##
      C
23      CALL OUTDSP
      C
      C INPUT HEX VALUE OF MEMORY BYTE TO DISPLAY
      C
24      CALL INPHX
      C
      C SAVE TEMP10 ADDRESS FOR AUTO INCREMENT
25      TEMP10=DUMMY(1)
      C VALUE PASSED IN TEMP10+1 SLOT
26      200  ADPSAV=TEMP10
      C
      C CALL AGETIT TO RECOVER CONTENTS OF MEMORY REFERENCED IN TEMP10
27      210  CALL AGETIT(TEMP10,DOBX)
      C
      C DOBX WILL HAVE TWO CONSECUTIVE BYTES, FORMAT AND DISPLAY
      C
28      CALL FORMAT(DOBX)
29      DOB(8)=DOB(6)
30      DOB(7)=DOB(5)
31      DOB(6)=#3DH
32      DOBZ=TEMP10
33      CALL FORMAT(DOBZ)
      C
34      CALL OUTDSP
      C
      C INPUT CONTROL VALUE FOR AUTO INCREMENT OR END
35      CALL INPCDU
      C
      C IF AUTO INCREMENT RECOVER ADDRESS INC BY 1 AND LOOP
      C NXT KEY
36      IF(NEXT) THEN
37          TEMP10=ADPSAV+1
38          GO TO 200
39      END IF
      C
      C ADJUST PRESSED, GET NEW VALUE
40      IF(ADJUST) THEN
41          DOBC=DEGMES(20)
42          CALL INPHX
      C DUMMY(1) = NEW BYTE
43          TEMP10=ADPSAV
44          CALL APUTIT(TEMP10)
45          GO TO 210
46      END IF
      C
      C ELSE RST , EXIT
47      RETURN
48      END

```

MODULE INFORMATION:

CODE AREA SIZE = 0004H 212D

VARIABLE AREA SIZE = 0002H 20
MAXIMUM STACK SIZE = 0000H 100
102 LINES READ

ORIGINAL PAGE IS
OF POOR QUALITY

0 PROGRAM ERROR(S) IN PROGRAM UNIT DEUGHE

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT PAUSE
 OBJECT MODULE PLACED IN F1:VCSIOU.OBJ
 COMPILER INVOKED BY FORT90 F1:VCSIOU.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE PAUSE
      C
      C PAUSE WAITS FOR THE USER TO PRESS ANY KEY ON THE INPUT C/D
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: INPCDU, OUTDSP
      C DESTROYS: NOTHING
      C MODIFIES: CPANEL
      C VERSION: 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*4 CIB(8), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
4      INTEGER*4 BITS(8), DUMMY(32)
      C
5      LOGICAL*4 NEXT, ADJUST, RESET, AUTO, MANUAL, FUN, CALERT, FIREFAL
      C
6      COMMON/CPANEL/ NEXT, ADJUST, RESET, AUTO, MANUAL, FIREFAL, FUN, CALERT
      C
7      COMMON/WORK/ BITS, DUMMY
      C
8      COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
9      SAVE /COMBUF/, /CPANEL/, /WORK/
      C
      C*****
      C
      C XXXXXXXX!
10     DOB(8)=#21H
      C
11     CALL OUTDSP
      C
12     10 CONTINUE
      C
13     CALL INPCDU
      C
14     IF(RESET OR NEXT OR ADJUST) GO TO 500
      C
15     GO TO 10
16     500 CONTINUE
17     RETURN
18     END

```

MODULE INFORMATION:

CODE AREA SIZE	= 001FH	31D
VARIABLE AREA SIZE	= 0000H	0D
MAXIMUM STACK SIZE	= 0002H	2D

FORTRAN COMPILER

PAGE 2

43 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT PAUSE

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II FORTRAN-80 V2.1 COMPILATION OF PROGRAM UNIT CLOCK
 OBJECT MODULE PLACED IN :F1:YCSIOU.OBJ
 COMPILER INVOKED BY: FORT80 :F1:YCSIOU.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE CLOCK
      C
      C CLOCK SYNCHRONIZES THE YCS TO THE TIMER UNIT
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: NOTHING
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION: 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*4 TFLAG(6)
      C
4      COMMON/CLOCK/TFLAG
      C
5      SAVE/CLOCK/
      C
      C*****
      C
6      I=0
      C TEST 1 SECOND FLAG BYTE FLAG FOR INTERRUPT PROCESS SYNC
7      100 IF(TFLAG(3).NE.0) GO TO 200
8      I=I+1
      C TIME OUT IF NO INTERRUPT
9      IF(I.GE.20000) GO TO 200
10     GO TO 100
11     200 CONTINUE
12     TFLAG(3)=0
      C
13     RETURN
14     END

```

MODULE INFORMATION:

```

/ CODE AREA SIZE      = 0035H    53D
\ VARIABLE AREA SIZE = 0062H    2D
  MAXIMUM STACK SIZE = 0002H    2D
  33 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT CLOCK

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT INPCDU
 OBJECT MODULE PLACED IN :F1:VCSIOU.OBJ
 COMPILER INVOKED BY: FORT90 :F1:VCSIOU.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE INPCDU
      C
      C INPCDU READS THE C/D PANEL AND RETURNS A LOGICAL ARRAY
      C
      C INPUTS: NONE
      C OUTPUTS: DATA IN /COMMON/
      C CALLS: AI0C0U
      C DESTROYS: NOTHING
      C MODIFIES: CONTROL WORDS
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 BITS(8), DUMMY(22)
      C
4      LOGICAL*1 LBITS(8), NEXT, ADJUST, RESET, AUTO, MANUAL, RUN, CALRT, PWRFL
      C
5      COMMON/CPANEL/NEXT, ADJUST, RESET, AUTO, MANUAL, PWRFL, RUN, CALRT
      C
6      COMMON/WORK/BITS, DUMMY
      C
7      COMMON/OPSVEC/OPMODE, MODECT, ENGMSE, ARRAY(7)
      C
8      COMMON/ERRFLG/ ERROR(8)
      C
9      SAVE /WORK/, /OPSVEC/, /ERRFLG/, /CPANEL/
      C
10     EQUIVALENCE (LBITS(1), NEXT)
      C
      C*****
      C
      C BITS(1), NEXT, ACTIVE LOW, 0
      C BITS(2), ADJUST, ACTIVE LOW, 0
      C BITS(3), RUN/CAL, RUN ACTIVE HIGH, 11111111B
      C          CAL ACTIVE LOW, 0
      C BITS(4), RESET, ACTIVE LOW, 0
      C BITS(5), AUTO, ACTIVE LOW, 0
      C BITS(6), MANUAL, ACTIVE HIGH, 11111111B
      C BITS(7), POWER FAIL, ACTIVE HIGH, 11111111B
      C
      C CLEAR WORK STORAGE AND READ C/D PANEL
      C
11     DO 100 I=1,8
12     BITS(I)=0
13     LBITS(I)=.FALSE.
14 100 CONTINUE
      C
15     CALL AI0C0U(BITS(1))
      C
      C TEST INPUTS AND RESET STATUS WORDS AS APPROPRIATE
      C

```

```

16      IF(BITS(1) EQ 0) THEN
17      C NEXT SWITCH PRESSED
18      NEXT= TRUE
19      END IF
20      C
21      IF(BITS(2) EQ 0) THEN
22      C ADJUST SWITCH PRESSED
23      ADJUST= TRUE
24      END IF
25      C
26      IF(BITS(3) EQ 0) THEN
27      C CALIBRATE MODE SWITCH SET
28      C IF IN RUN MODE NOW EXIT TO HALT
29      CALIB= TRUE
30      C IF IN AUTO OR MANUAL MODES
31      IF((OPMODE EQ 1) OR (OPMODE EQ 4)) THEN
32      OPMODE=3
33      RETURN
34      END IF
35      C
36      END IF
37      C
38      IF(BITS(4) EQ 0) THEN
39      C RESET SWITCH PRESSED
40      RESET= TRUE
41      END IF
42      C
43      C IF RUN SWITCH ON
44      C
45      IF(BITS(5) NE 0) THEN
46      C
47      RUN= TRUE
48      IF(BITS(6) EQ 0) THEN
49      C AUTOMATIC MODE SWITCH PRESSED
50      AUTO= TRUE
51      OPMODE=1
52      C RESET TIME OUT ON CONTROL
53      ZERO=0
54      CALL AIOVOT(ZERO)
55      END IF
56      C
57      IF(BITS(6) NE 0) THEN
58      C MODE = MANUAL DISABLE SYSTEM
59      MANUAL= TRUE
60      OPMODE=4
61      END IF
62      C
63      END IF
64      C
65      IF(BITS(7) NE 0) THEN
66      C POWER FAILURE OR SHUT DOWN ON 39 VOLT CONTROL VOLTAGE
67      C HALT SYSTEM
68      PWRFAL= TRUE
69      OPMODE=0
70      ERROR(2)=1
71      C
72      END IF

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      C
50      RETURN
51      END

```

ORIGINAL PAGE IS
OF POOR QUALITY

MODULE INFORMATION:

```

CODE AREA SIZE      = 011CH      2840
VARIABLE AREA SIZE = 0004H       40
MAXIMUM STACK SIZE = 0004H       40
111 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT INPCDU

ISIS-II FORT800 V2.1 COMPILATION OF PROGRAM UNIT INPMEX
 OBJECT MODULE PLACED IN :F1:VCSIOU.OBJ
 COMPILER INVOKED BY: FORT800 :F1:VCSIOU.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE INPMEX
      C
      C INPMEX GETS A HEX NUMBER FROM THE C/D BY DYNAMIC PROMPTING
      C
      C INPUTS: NONE
      C OUTPUTS: HEX VALUE IN TEMP10
      C CALLS: AIOCDU, OUTDSP, FORMAT
      C DESTROYS: NOTHING
      C MODIFIES: /WORK/, DOB
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 WORK(8), XDUM(8)
4      INTEGER*1 CIB(80), TOB(128), PIB(2), POB(2), KIB(8), VOB(2)
      C
5      DIMENSION DNUM(4)
      C
6      CHARACTER*1 DOB(8)
      C
7      COMMON/WORK/TEMP10, DNUM, MAG, ICNT, ICOL, NUM, I, J, WORK, DUMMY(5)
      C
8      COMMON/COMBUF/CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
9      SAVE /COMBUF/, /WORK/
      C
10     EQUIVALENCE (DOB(5), DOBX), (DOB(1), XDUM(1))
      C
      C*****
      C
      C INITIALIZE PARAMETERS AND CLEAR WORK AREA
      C
11     DO 100 I=1, 4
12     DNUM(I)=0
13     100 CONTINUE
      C
14     DOBX=0
15     MAG=1
16     NUM=0
17     ICOL=0
18     ICNT=0
19     GO TO 2150
      C
20     1000 CONTINUE
21     CALL AIOCDU (WORK(1))
      C
      C TEST IF RESET (ENTER) KEY PRESSED
      C
22     IF (WORK(4) EQ 0) GO TO 3000
      C
      C TEST IF NEXT KEY PRESSED
  
```

ORIGINAL PAGE IS
OF POOR QUALITY

```

23      IF(WORK(1).EQ.0) THEN
24          DNUN=ICOL+1)*ICNT
25          ICOL=MOD(ICOL+L,4)
26          ICNT=0
27          MAG=16**ICOL
28          GO TO 2000
29      END IF

      C
      C TEST IF ADJUST KEY PRESSED
      C
30      IF(WORK(2).EQ.0) THEN
31          ICNT=MOD(ICNT+L,16)
32          DNUN(ICOL+1)=ICNT
33          GO TO 2000
34      END IF

      C
      C BUILD VALUE FROM INPUTS
      C
35      2000 DOBX=0
36          J=1
37          DO 2100 I=L,4
38              DOBX=DOBX+DNUN(I)*J
39              J=J*16
40      2100 CONTINUE
41      2150 CONTINUE

      C
42      TEMP10=DOBX

      C
43      CALL FORMAT(DOBX)

      C
      C # IN UNSET DIGITS
44      IF(ICOL.LT.3) THEN
45          J=3-ICOL
46          DO 2200 I=L,J
47              XDUN(I+4)=#23H
48      2200 CONTINUE
49      END IF
50      CALL OUTDSP

      C
51      GO TO 1000

      C
52      2000 CONTINUE
53      DNUN(1)=TEMP10
54      TEMP10=#FH

      C
55      RETURN
56      END

```

MODULE INFORMATION:

```

CODE AREA SIZE      = 017DH      281D
VARIABLE AREA SIZE  = 0002H      2D
MAXIMUM STACK SIZE  = 0002H      2D
98 LINES READ

```


FORTRAN COMPILER

PAGE 9

@ PROGRAM ERROR(S) IN PROGRAM UNIT INPHEN

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT PROMPT
 OBJECT MODULE PLACED IN F1:VCS10U.OBJ
 COMPILER INVOKED BY: FORT90 F1:VCS10U.FOP

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE PROMPT
      C
      C   PROMPT DISPLAYS A PROMPT IN LEDS AND INPUTS C/D
      C
      C   INPUTS: NONE
      C   OUTPUTS: NONE
      C   CALLS: INPCDU, ATODSP
      C   DESTROYS: NOTHING
      C   MODIFIES: NOTHING
      C   VERSION: 1.0
      C
2      INTEGER*4 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
      C
3      COMMON/COMBUF/CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
4      SAVE /COMBUF/
      C
      C*****
      C
5      CALL ATODSP(DOB(1))
      C
6      CALL INPCDU
      C
7      RETURN
8      END
  
```

MODULE INFORMATION:

CODE AREA SIZE = 000AH 100
 VARIABLE AREA SIZE = 0000H 00
 MAXIMUM STACK SIZE = 0002H 20
 25 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT PROMPT
 0 TOTAL PROGRAM ERROR(S)
 END OF FORTRAN COMPILATION

ISIS-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT CONFNC
 OBJECT MODULE PLACED IN :F1:VCECON.OBJ
 COMPILER INVOKED BY: FORT80 :F1:VCECON.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE CONFNC
      C
      C CONFNC IS A TABLE DRIVEN ROUTINE WHICH IS USED TO CONVERT
      C     SENSOR DATA TO ENGINEERING UNITS AS A+B*X=Y
      C
      C INPUTS: SENSOR INDEX AND DATA
      C OUTPUTS: NONE
      C CALLS: RANGCK (OPTION)
      C DESTROYS: NOTHING
      C MODIFIES: TEMP10
      C VERSION: 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      DIMENSION SENTAB(10,2),RGCDAT(10,2)
      C
4      COMMON/WORK/ TEMP10,SINFNT,DUMMY(10)
      C
5      COMMON/CONTAB/SENTAB,RGCDAT
      C
6      COMMON/OPSVEC/ OPMODE,MODECT,RNGMSK,ARRAY(7)
      C
7      SAVE /CONTAB/,/OPSVEC/,/WORK/
      C
      C*****
      C
      C CONVERT VALUE IN TEMP10 TO ENGINEERING UNITS Y = A+B*X
      C
8      TEMP10=SENTAB(SINFNT,1)+TEMP10*SENTAB(SINFNT,2)
      C
      C TEST IF RANGE CHECK NEEDED
      C
9      IF(RNGMSK.NE.0) THEN
10         CALL RANGCK
11     END IF
      C
12     RETURN
13     END

```

MODULE INFORMATION:

CODE AREA SIZE	= 003AH	58D
VARIABLE AREA SIZE	= 0000H	00
MAXIMUM STACK SIZE	= 0004H	40
38 LINES READ		

0 PROGRAM ERROR(S) IN PROGRAM UNIT CONFNC

ISIS-11 FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT RANGCK
 OBJECT MODULE PLACED IN :F1:VCSOON.OBJ
 COMPILER INVOKED BY: FORT90 :F1:VCSOON.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE RANGCK
      C
      C RANGCK TESTS CONVERTED SENSOR DATA AGAINST MAX AND MIN LIMITS
      C
      C INPUTS: INDEX POINTER, DATA
      C OUTPUTS: ERROR FLAG 1
      C CALLS: NOTHING
      C DESTROYS: NOTHING
      C MODIFIES: SDB
      C VERSION: 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      DIMENSION RGCDAT(10,2), SENTAB(10,2)
      C
4      COMMON/CONTAB/ SENTAB, RGCDAT
      C
5      COMMON/ERRFLG/ ERROR(S)
      C
6      COMMON/WORK/ TEMP10, SINPNT, DUMMY(18)
      C
7      SAVE /WORK/, /CONTAB/, /ERRFLG/
      C
      C*****
      C
      C TEST DATA AGAINST MAX AND PROCESS
      C
9      IF(TEMP10.GT.RGCDAT(SINPNT,2)) THEN
10     TEMP10=RGCDAT(SINPNT,2)
11     ERROR(1)=SINPNT
12     RETURN
      C
      C TEST DATA AGAINST MINIMUM AND PROCESS
      C
13     IF(TEMP10.LT.RGCDAT(SINPNT,1)) THEN
14     TEMP10=RGCDAT(SINPNT,1)
15     ERROR(1)=10+SINPNT
16     RETURN
17     END IF
      C
18     RETURN
19     END
  
```

MODULE INFORMATION:

CODE AREA SIZE	= 0064H	1000
VARIABLE AREA SIZE	= 0000H	00
MAXIMUM STACK SIZE	= 0002H	20

FORTRAN COMPILER

PAGE 3

43 LINES READ

ORIGINAL PAGE IS
OF POOR QUALITY

0 PROGRAM ERROR(S) IN PROGRAM UNIT RANGE

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ORIGINAL PAGE 1:
OF POOR QUALITY

ISIS-II F0RTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT INPCID
OBJECT MODULE PLACED IN :F1:VCSINP.OBJ
COMPILER INVOKED BY: FORT83 :F1:VCSINP.FOR

```

1      SUBROUTINE INPCID
      C
      C INPCID INPUTS THE CID DATA VIA PORT 01, CONTROL VIA PORT 02
      C
      C INPUTS: NONE
      C OUTPUTS: CID DATA IN SIB
      C CALLS: @AIOCID
      C DESTROYS: NOTHING
      C MODIFIES: SIB
      C VERSION: 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      DIMENSION SIB(10), SDB(10)
      C
4      COMMON/SENBUF/ SIB, SDB
      C
5      COMMON/WORK/ TEMP10, SINPNT, DUMMY(10)
      C
6      SAVE /SENBUF/, /WORK/
      C
      C*****
      C
      C SET SINPNT TO CID VALUE
      C
7      SINPNT=6
      C
8      CALL AIOCID (TEMP10)
      C
9      SIB(SINPNT)=TEMP10
      C
10     RETURN
11     END

```

MODULE INFORMATION:

CODE AREA SIZE = 001EH 30D
VARIABLE AREA SIZE = 0000H 0D
MAXIMUM STACK SIZE = 0002H 2D
33 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT II:PCID

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT INPPKS
 OBJECT MODULE PLACED IN :F1:VCSINP.OBJ
 COMPILER INVOKED BY: FORT90 :F1:VCSINP.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE INPPKS
      C
      C INPPKS INPUTS THE FRONT AND REAR PICK SIGNALS VIA PORTS 11,13/CNT 10,12
      C
      C INPUTS: NONE
      C OUTPUTS: PICK DATA IN SIB
      C CALLS: INPUT (FORTRAN ROUTINE)
      C DESTROYS: NOTHING
      C MODIFIES: SIB
      C VERSION: 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 I
      C
4      DIMENSION SIB(10),SDB(10)
      C
5      COMMON/SENBUF/ SIB,SDB
      C
6      COMMON/WORK/TEMP10,SINPNT,DUMMY(10)
      C
7      SAVE /SENBUF/,/WORK/
      C
      C*****
      C
8      CALL INPUT(10H,I)
      C INPUT PORT WITH 4 PICK BITS
      C
9      I=I.AND.#0FH
      C
10     SIB(7)=I.AND.#03H
      C
11     SIB(8)=I.AND.#0CH
      C
12     IF(SIB(7).GT.0) SIB(7)=1
      C
13     IF(SIB(8).GT.0) SIB(8)=1
      C
14     RETURN
15     END
  
```

MODULE INFORMATION:

CODE AREA SIZE = 0046H 72D
 VARIABLE AREA SIZE = 0001H 1D
 MAXIMUM STACK SIZE = 0002H 2D
 40 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT INPPKS

FORTRAN COMPILER

PAGE 3

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II FORTAN-90 V2.1 COMPILATION OF PROGRAM UNIT OUTCAS
 OBJECT MODULE PLACED IN :F1:VCSIOZ.GBJ
 COMPILER INVOKED BY: FORT60 :F1:VCSIOZ.FOR

ORIGINAL PAGE IS
 OF POOR QUALITY

```

1      SUBROUTINE OUTCAS
      C
      C OUTCAS OUTPUTS THE CASSETTE OUTPUT BUFFER TO THE DIGITAL CASSETTE
      C     VIA PORT 00,09,0A, CONTROL PORT 0B
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: AIOCAS
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
4      INTEGER*1 DUMMY(2), IOFVEC(20), TFLAG(6), CASNRK(4)
      C
5      LOGICAL*1 IOFCAS
      C
6      COMMON/ERRFLG/ERROR(8)
      C
7      COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
8      COMMON/IOFVEC/IOFVEC
      C
9      COMMON/OPSPNT/OPPNTS(8)
      C
10     COMMON/CLOCK/TFLAG
      C
11     DIMENSION CASDAT(10)
      C
12     EQUIVALENCE (CASNRK(1),TEMP10), (CASNRK(3),DUMMY(1))
13     EQUIVALENCE (CASNRK(4),DUMMY(2))
14     EQUIVALENCE (CASDAT(1),TOB(1))
15     EQUIVALENCE (IOFCAS,IOFVEC(15))
      C
16     SAVE /ERRFLG/, /OPSPNT/, /COMBUF/, /IOFVEC/, /CLOCK/
      C
      C*****
      C
      C FORMAT OF CASSETTE CONTROL BUFFER
      C TO AIOCAS IO DRIVER
      C
      C CASNRK(4) INTEGER*1
      C TEMP10,DUMMY(1),DUMMY(2)
      C DATA  Cntl  ERROR
      C WORD  BYTE  BYTE
      C
      C CONTROL =1 WRITE; =2 FAST FORWARD
      C
      C ERROR => 0K ; =1 BEOT
  
```

ORIGINAL PAGE IS
OF POOR QUALITY

```

C
C OPPNTS(2) = POINTER TO ENTPV
C OPPNTS(3) = NUMBER OF WORDS IN BLOCK
C
C *****
C
17      TFLAG(2)=0
19      IF(10FCAS) THEN
C TEST IF END OF BLOCK 64 WORDS
19      IF(OPPNTS(3) GE 64) GO TO 100
C TEST IF DATA LEFT IN BUFFER
C IF 0 DONE SKIP ELSE WRITE DATA
20      IF(OPPNTS(2) GT 0) THEN
C
21      DUMMY(1)=1
22      DUMMY(2)=0
23      TEMP10=CASDAT(OPPNTS(2))
C
C CALL CASSETTE IO DRIVER ROUTINE
24      CALL AIOCRS(TEMP10)
C
C TEST IF ERROR CONDITION RETURNED BY AIOCRS
25      IF(DUMMY(2) NE 0) ERROR(5)=DUMMY(2)
26      OPPNTS(2)=OPPNTS(2)-1
27      OPPNTS(3)=OPPNTS(3)+1
C
29      END IF
C END SECOND IF , (OPPNTS)
29      END IF
C END FIRST IF , (10FCAS)
30      RETURN
C
C RESET POINTER FOR DATA BUFFER; WAIT ONE 100MSEC INTERRUPT
31 100 OPPNTS(3)=0
32      RETURN
33      END

```

MODULE INFORMATION:

```

CODE AREA SIZE      = 0074H    116D
VARIABLE AREA SIZE = 0004H     4D
MAXIMUM STACK SIZE = 0002H     2D
87 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT OUTCAS

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT OUTDSP
OBJECT MODULE PLACED IN :F1:VCSIOZ.OBJ
COMPILER INVOKED BY: FORT80 :F1:VCSIOZ.FOR

```
1      SUBROUTINE OUTDSP
      C
      C OUTDSP OUTPUTS THE DISPLAY BUFFER TO THE LED DISPLAY UNIT
      C
      C INPUTS: NONE
      C OUTPUTS: NONE
      C CALLS: AICDSP
      C DESTROYS: NOTHING
      C MODIFIES: NOTHING
      C VERSION 1.0
      C
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*1 CIB(80), KIB(8), DOB(9), TOB(129), PIB(2), POB(2), VOB(2)
      C
4      COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
5      SAVE /COMBUF/
      C
      C*****
      C
6      CALL AICDSP(DOB(1))
      C
7      RETURN
8      END
```

MODULE INFORMATION:

CODE AREA SIZE = 0007H 7D
VARIABLE AREA SIZE = 0000H 00
MAXIMUM STACK SIZE = 0002H 2D
25 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT OUTDSP

ISIS-II FORTRAN-90 V2.1 COMPILATION OF PROGRAM UNIT SAVDAT
 OBJECT MODULE PLACED IN :F1:VCSIOZ.GBJ
 COMPILER INVOKED BY: FORT90 :F1:VCSIOZ.FOR

```

1      SUBROUTINE SAVDAT
      C
      C SAVDAT LOADS DATA TO BE STORED ON CASSETTE INTO BUFFER
      C AND SETS UP POINTERS
      C
      C INPUTS: NONE
      C OUTPUTS: TAPE OUTPUT BUFFER AND POINTERS
      C CALLS: NOTHING
      C DESTROYS: NOTHING
      C MODIFIES: TOB, POINTERS
      C VERSION 1.0
      C .
2      IMPLICIT INTEGER*2 (A-Z)
      C
3      INTEGER*4 CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), VOB(2)
4      INTEGER*4 TFLAG(6)
      C
5      COMMON/CLOCK/TFLAG
      C
6      COMMON/COMBUF/ CIB, KIB, DOB, TOB, PIB, POB, VOB
      C
7      COMMON/SENBUF/ SIB(10), SDB(10)
      C
8      COMMON/OPSPNT/OPPNTS(8)
      C
9      COMMON/PARAMS/PARMS(20)
      C
10     COMMON/OPSEVC/OPHODE, MODECT, ODUN(8)
      C
11     DIMENSION CASDAT(10)
      C
12     EQUIVALENCE (CASDAT(1), TOB(1)), (PARMS(17), ENDETP)
      C
13     SAVE /OPSPNT/, /PARAMS/, /OPSEVC/
14     SAVE /COMBUF/, /SENBUF/, /OPSPNT/, /SENMEM/, /CLOCK/
      C
      C*****
15     IF(OPPNTS(2).NE.0) GO TO 1000
      C
      C LOAD SENSOR DATA (5 WORDS) INTO BUFFER
16     DO 100 I=2,5
17     CASDAT(I)=SDB(I-1)
18     CONTINUE
      C
      C FORM = 111111XXYY111111 FOR MARKING RECORD
19     CASDAT(1)=#0FC3FH
      C CLEAR CASDAT(1) AND PACK 4 NUMBERS (NISELES) INTO IT
      C
20     IF(VOB(2).NE.0) CASDAT(1)=CASDAT(1).OR.#0040H
      C

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

21      IF(V08(1).NE.0) CASDAT(1)=CASDAT(1).OR.#0080H
      C
22      IF(SDB(3).NE.0) CASDAT(1)=CASDAT(1).OR.#0180H
      C
23      IF(SDB(7).NE.0) CASDAT(1)=CASDAT(1).OR.#0200H
      C
      C FORM = 111100XXV001111 FOR MARKING RECORDS
24      IF(OPMODE.EQ.1) THEN
25          TEMPX=#0280H
26      ELSE
27          TEMPX=#0180H
28      END IF
      C
29      IF(MODECT.EQ.1) THEN
30          TEMPY=#0080H
31      ELSE
32          TEMPY=#0040H
33      END IF
      C
34      CASDAT(7)=#0F00FH+TEMPX+TEMPY
      C
35      TFLAG(2)=0
      C
      C SET COUNTER FOR DATA WORDS TO WRITE
36      OPNTS(2)=7
      C
37      1800 CONTINUE
      C
38      RETURN
39      END

```

MODULE INFORMATION:

```

CODE AREA SIZE    = 0182H    2580
VARIABLE AREA SIZE = 0006H     60
MAXIMUM STACK SIZE = 0002H    20
31 LINES READ

```

0 PROGRAM ERROR(S) IN PROGRAM UNIT SAYCAT

```

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

```

1515-11 FORTSAP-90 V2.1 COMPILATION OF PROGRAM UNIT TAPITZ
OBJECT MODULE PLACED IN F1:VCSAP OBJ
COMPILED INVOKED BY: FORT90 F1:VCSAP FOR

```

1      SUBROUTINE TAPITZ
      C
      C TAPITZ WRITES THE TAPE HEADER
      C
2      $INCLUDE('F1:VCSNEM.FOR')
      C *****
      C
      C VCS SYSTEM MEMORY ALLOCATION
      C
3      =      IMPLICIT INTEGER*2(A-Z)
4      =      INTEGER*1 CIB,KIB,COB,TOB,PIS,POB,YOB
5      =      INTEGER*1 PKFBUF,PKRBUF,PKFSAY,PKPSAY
6      =      INTEGER*1 NMXPX(6),TFLAG(6)
7      =      INTEGER*1 ERRBUF(8,10)
      C
8      =      LOGICAL*1 IOFVEC(20),CNTVEC(16),SINT(8),SMEM(8),DEGM(8)
9      =      LOGICAL*1 DEFIOV(20),DEFCTV(16)
10     =      LOGICAL*1 NEXT,ADJUST,RESET,RUN,CALENT,PHRFAL,AUTO,MANUAL
      C
11     =      CHARACTER*4 PARMES,MOSMES(10)
12     =      CHARACTER*1 DSPCOD(10)
13     =      CHARACTER*4 DOEC,DOEN,OPSMES(10)
14     =      CHARACTER*4 ERRMES(20),DEGMES(20),RUNMES(20)
      C
15     =      DIMENSION DEFPRM(20),DEFPNT(8)
      C
      C COMMON BLOCK ALLOCATIONS
      C
16     =      COMMON/CPANEL/NEXT,ADJUST,RESET,AUTO,MANUAL,PHRFAL,RUN,CALENT
      C
17     =      COMMON/MESTB2/ PARMES(20)
      C
18     =      COMMON/MESTB3/OPSMES
      C
19     =      COMMON/MOSMES/MOSMES
      C
20     =      COMMON/SENPNT/SENPTR(16)
      C
21     =      COMMON/PIKBUF/PKFBUF(10),PKRBUF(10),PKFSAY,PKPSAY
      C
22     =      COMMON/CLOCK/TFLAG
      C
23     =      COMMON/DEBUG/DBGM
      C
24     =      COMMON/SENDEL/LCDEL(40),PCDEL(60),FDRDEL(400)
      C
25     =      COMMON/IOFVEC/IOFVEC
      C
26     =      COMMON/CNTVEC/CNTVEC
      C
27     =      COMMON/OPSVEC/OPSVEC(10)

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

      = C
29 = COMMON/OPPARM/ICMNT(10), NMAX(16), NMXIPX
      = C
29 = COMMON/SENMEM/LSTPAS(3000), DSCNTL(16)
      = C
30 = COMMON/OPSPNT/OPPNTS(8)
      = C
31 = COMMON/WORK/TEMP10, DUMMY(19)
      = C
32 = COMMON/SAVER/SAVE(20)
      = C
33 = COMMON/PARAMS/ PARAM(20)
      = C
34 = COMMON/ERRBUF/ERRBUF
      = C
35 = COMMON/ERRFLG/ ERROR(8)
      = C
36 = COMMON/COMBUF/CIB(80), KIB(8), DOB(8), TOB(128), PIB(2), POB(2), YOB(2)
      = C
37 = COMMON/SENBUF/SIB(10), SOB(10)
      = C
38 = COMMON/CONTAB/SENTAB(10, 2), RECDAT(10, 2)
      = C
39 = COMMON/MESTAB/ERRMES, DBGMES, RUNMES, DSPCCO
      = C
40 = COMMON/DEFAULT/ DEFIOV, DEFPRM, DEFCTV, DEFINT
      = C
41 = COMMON/DEFTAB/ DEFRNG(10, 2), DEFCON(10, 2)
      = C
42 = COMMON/DEFOPS/DEFOPS(10)
      = C
43 = COMMON/DEFMOS/DEFMOS(16)
      = C
44 = SAVE /CPANEL/, /PIKBUF/, /SAVER/, /ERRBUF/, /SENDEL/
45 = SAVE /DEBUG/, /IOFVEC/, /CNTVEC/, /OPSVEC/
46 = SAVE /OPPARM/, /SENMEM/, /OPSPNT/, /CLOCK/, /WORK/, /DEFTAB/
47 = SAVE /PARAMS/, /ERRFLG/, /COMBUF/, /SENBUF/, /CONTAB/, /MESTAB/, /DEFAULT/
48 = SAVE /SENPT/, /MESTB2/, /MESTB3/, /MEFOPS/
49 = SAVE /DEFMOS/
      = C
50 = EQUIVALENCE (DOB(1), DOBC), (DOB(5), DOBN)
51 = EQUIVALENCE (OPSVEC(1), OPMODE), (OPSVEC(2), MODECT), (OPSVEC(3), ENGMEK)
52 = EQUIVALENCE (OPSVEC(4), OPSENR), (OPSVEC(5), RATECK)
53 = EQUIVALENCE (OPSVEC(6), STATUS), (TIME, TFLAG(4))
      = C
      C
54      INTEGER*1 CNTL(4)
      C
55      DIMENSION CASHRK(8)
      C
56      EQUIVALENCE(CNTVEC(1), CASHRK(1))
57      EQUIVALENCE (CNTL(1), DUMMY(1))
      C
      C*****
      C
      C IF CASSETTE ON, WRITE HEADER FILE
      C

```

ORIGINAL PAGE IS
OF POOR QUALITY

```

58      DOBN=DOBNES(19)
59      CALL OUTDSP
      C WRITE FILE GAP TO START
60      CRLF=#0ACDH
61      CNTL(2)=0
62      CNTL(1)=1
      C WRITE A CARRIAGE RETURN LINE FEED
63      TEMP10=CRLF
64      CALL AIOCRS(TEMP10)
65      CALL MSEC2H
      C
      C WRITE F6C5H PATTERN
66      TEMP10=#0F6C5H
67      DO 1800 I=1,20
68      CALL AIOCRS(TEMP10)
69      CALL MSEC2H
70  1800 CONTINUE
      C
71      TEMP10=CRLF
72      CALL AIOCRS(TEMP10)
73      CALL MSEC2H
      C
      C WRITE 20 WORDS THEN CR/LF
74      DO 2000 I=1,20
75      TEMP10=PARAM(I)
76      CALL AIOCRS(TEMP10)
77      CALL MSEC2H
78  2000 CONTINUE
      C
79      TEMP10=CRLF
80      CALL AIOCRS(TEMP10)
81      CALL MSEC2H
      C WRITE OPERATION PARAMS
82      DO 1900 I=1,10
83      TEMP10=OPSVEC(I)
84      CALL AIOCRS(TEMP10)
85      CALL MSEC2H
86  1900 CONTINUE
      C
87      DO 1950 I=1,10
88      TEMP10=CASHRK(I)
89      CALL AIOCRS(TEMP10)
90      CALL MSEC2H
91  1950 CONTINUE
      C
92      TEMP10=CRLF
93      CALL AIOCRS(TEMP10)
94      CALL MSEC2H
      C
95      RETURN
96      END

```

MODULE INFORMATION:

CODE AREA SIZE = 012AH 2980

FORTRAN COMPILER

ORIGINAL PAGE IS
OF POOR QUALITY

PAGE 4

VARIABLE AREA SIZE = 0014H 200
MAXIMUM STACK SIZE = 0000H 100
159 LINES READ

0 PROGRAM ERROR(S) IN PROGRAM UNIT TAPITZ

0 TOTAL PROGRAM ERROR(S)
END OF FORTRAN COMPILATION

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	
		2 NAME VCSUTL	
		3 ;	
		4 PUBLIC FORMAT, A100SP, APUTIT	
		5 ;	
		6 EXTRN HEXASC, OUTPT	
		7 ;	
		8 CSEG	
		9 ;	
		10 ;*****	
		11 ;	
		12 ; SUBROUTINE A100SP	
		13 ;	
		14 ; A100SP IS THE DISPLAY DRIVER INTERFACE FOR THE VCS	
		15 ;	
		16 ;	
		17 ; INPUTS: ADDRESS OF DOB(1) IN BC PAIR	
		18 ; OUTPUTS: NONE	
		19 ; CALLS: MONITR:OUTPT	
		20 ; DESTROYS: F/F'S	
		21 ; MODIFIES: NOTHING	
		22 ; VERSION 1.0	
		23 ;	
		24 A100SP:	
0000	C5	25 PUSH B ; GET BC ONTO STACK	
0001	E1	26 POP H ; POP IT INTO HL PAIR FOR MEMORY REFERENCE	
0002	0E08	27 MVI C,08H ; SET CHARACTER COUNT OF 9 IN C	
0004	CD0000	28 CALL OUTPT ; CALL DISPLAY DRIVER IN MONITOR	E
0007	C1	29 POP B ; RECOVER BC PAIR	
0008	C9	30 RET ; RETURN TO FORTRAN CALLING ROUTINE	
		31 ;	
		32 ;*****	
		33 ;	
		34 ; SUBROUTINE FORMAT	
		35 ;	
		36 ; FORMAT TURNS 2 HEX BYTES IN DOB(5,6) INTO ASCII IN DOB(5,6,7,8)	
		37 ;	
		38 ; INPUTS: ADDRESS OF DOB(5) IN BC PAIR	
		39 ; OUTPUTS: ASCII CODE IN DOB(5,6,7,8)	
		40 ; CALLS: MONITR:HEXASC	
		41 ; DESTROYS: A, F/F'S, H, L	
		42 ; MODIFIES: NOTHING	
		43 ; VERSION: 1.0	
		44 ;	
		45 FORMAT:	
0009	C5	46 PUSH B ; PUSH BC ONTO STACK	
000A	E1	47 POP H ; POP ADDRESS INTO HL PAIR	
000B	23	48 INX H ; SET HL TO POINT TO DOB(8)	
000C	23	49 INX H	
000D	23	50 INX H	
000E	C5	51 PUSH B ; SAVE BC ADDRESS OF DOB(5) ONTO STACK	
000F	CD1900	52 CALL FORM05 ; CALL DECODING ROUTINE	C
0012	C1	53 POP B ; GET BACK POINTER TO DOB(5)	
0013	03	54 INX B ; SET BC TO SECOND BYTE	

LOC	OBJ	LINE	SOURCE STATEMENT
0014	2B	55	DCX H ; MOVE HL -1 TO DOB(E)
0015	CD1900	C 56	CALL FORM05 ; CALL DECODING ROUTINE
0018	C9	57	RET ; RETURN TO FORTAN
		58 ;	
		59	FORM05:
0019	0A	60	LDAX B ; GET # (BC)
001A	57	61	MOV D,A ; SAVE IN D
001B	E50F	62	ANI 0FH ; KEEP LOWER NIBBLE
001D	CD0000	E 63	CALL HEXASC ; CALL MONITOR HEX TO ASCII
0020	77	64	MOV M,A ; STORE ASCII INTO DOB
0021	2B	65	DCX H ; MOVE HL -1 IN DOB
0022	7A	66	MOV A,D ; GET FROM D
0023	0F	67	RRC
0024	0F	68	RRC ; KEEP UPPER NIBBLE
0025	0F	69	RRC
0026	0F	70	RRC
0027	CD0000	E 71	CALL HEXASC ; CALL MONITOR HEX TO ASCII ROUTINE
002A	77	72	MOV M,A ; STORE ASCII (HL)
002B	C9	73	RET ; RETURN TO FORMAT CALL
		74 ;	
		75 ;	*****
		76 ;	
		77 ;	SUBROUTINE APUTIT
		78 ;	
		79 ;	APUTIT PUTS A BYTE INTO MEMORY FROM FORTAN (POKE)
		80 ;	
		81 ;	INPUTS: BC -> TEMP0=ADDR,BYTE
		82 ;	OUTPUTS: NONE
		83 ;	CALLS: NOTHING
		84 ;	DESTROYS: A,F/F'S,BC
		85 ;	MODIFIES: MEMORY AT ADDR
		86 ;	VERSION: 1.0
		87 ;	
		88	APUTIT:
002C	0A	89	LDAX B ; GET ADDR LSB
002D	0F	90	MOV L,A
002E	03	91	INX B ; POINT TO MSB
002F	0A	92	LDAX B ; GET MSB
0030	67	93	MOV H,A
		94 ;	HL -> MEMORY PATCH ADDR
0031	03	95	INX B ; BC -> BYTE
0032	0A	96	LDAX B ; ACC = BYTE
0033	77	97	MOV M,A ; SET BYTE IN RAM
0034	C9	98	RET
		99 ;	
		100	END

PUBLIC SYMBOLS

RIODSP C 0000 APUTIT C 002C FORMAT C 0009

EXTERNAL SYMBOLS

HEXASC E 0000 OUTPT E 0000

USER SYMBOLS

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 9000/9005 MACRO ASSEMBLER V3.0 YCUTL PAGE 2

ATCOSP C 0000 AFUTIT C 0020 FORM05 C 0019 FORMAT C 0009 HEXASC E 0000 OUTPT E 0000

ASSEMBLY COMPLETE. NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	
		2	NAME AI010X
		3 ;	
		4	PUBLIC AI0CID, AI0VOT, AI0CRS
		5 ;	
		6	EXTRN MSEC2H
		7 ;	
		8	CSEG
		9 ;	
		10 ;	*****
		11 ;	
		12 ;	SUBROUTINE AI0CID
		13 ;	
		14 ;	AI0CID IS THE INPUT DRIVER FOR THE CID UNIT
		15 ;	
		16 ;	INPUTS: ADDRESS OF TEMP10 IN BC
		17 ;	OUTPUTS: CID DATA IN TEMP10
		18 ;	CALLS: NOTHING
		19 ;	DESTROYS: A, F, F'S
		20 ;	MODIFIES: TEMP10
		21 ;	VERSION 1.0
		22 ;	
		23	AI0CID:
		24 ;	GET ADDR BC - TEMP10
0000	69	25	MOV L, C
0001	68	26	MOV H, B
0002	0833	27	IN 033H ; LATCH COUNT
0004	0832	28	IN 032H ; RESET TO START COUNT
0005	0831	29	IN 031H ; INPUT MSB
0008	47	30	MOV B, A ; PUT MSB INTO B
0009	0830	31	IN 030H ; GET LSB
000B	4F	32	MOV C, A ; PUT INTO C
		33 ;	HL NOW POINTS TO TEMP10 FOR STORE
000C	71	34	MOV M, C ; PUT LSB AT TEMP10
000D	23	35	INX H ; HL=HL+1
000E	70	36	MOV M, B ; PUT MSB AT TEMP10 +1
		37 ;	
000F	09	38	RET ; RETURN TO FORTRAN
		39 ;	
		40 ;	*****
		41 ;	
		42 ;	SUBROUTINE AI0VOT
		43 ;	
		44 ;	AI0VOT IS THE VALVE DRIVER IO ROUTINE
		45 ;	
		46 ;	INPUTS: ADDRESS OF CBYTE IN BC
		47 ;	OUTPUTS: VALVE CONTROL BITS TO PORT 2, RESET+LEDS
		48 ;	CALLS: NOTHING
		49 ;	DESTROYS: A, F, F'S
		50 ;	MODIFIES: TEMP10
		51 ;	VERSION 1.0
		52 ;	
		53	AI0VOT:
0010	3E0F	54	NYI A, 0FH ; CLEAR ACC - OUTPUT = 00001111

ORIGINAL PAGE IS
OF POOR QUALITY

IS15-IT 8800/2085 MACRO ASSEMBLER, V3.0

A1010X PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
0012	D306	55	OUT 06 ; SEND TO CLEAR PORT BITS
0014	C5	56	PUSH B ; WASTE TIME
0015	C1	57	POP B
0016	3E2F	58	MVI A,02FH ; LOAD 1 BIT TO RESET LINE FOR ONE SHOT
0018	D306	59	OUT 06 ; SEND BIT TO UNIT - OUTPUT 00101111
001A	C5	60	PUSH B ; WASTE TIME
001B	C1	61	POP B
001C	3E0F	62	MVI A,0FH ; SEND 00001111
001E	D306	63	OUT 06
0020	C5	64	PUSH B ; WASTE TIME
0021	C1	65	POP B
0022	3E1F	66	MVI A,01FH ; LOAD TRIGGER BIT
0024	D306	67	OUT 06 ; SEND 00011111
0026	C5	68	PUSH B ; WASTE TIME
0027	C1	69	POP B
0029	3E0F	70	MVI A,0FH ; CLEAR BITS
002A	D306	71	OUT 06 ; SEND 00001111
		72 ;	
002C	0A	73	LDAX B ; GET CBYTE FOR OUTPUT
002D	E60F	74	ANI 0FH ; KEEP LOW NIBBLE
002F	D303	75	OUT 03 ; TRANSMITT TO VALVE DRIVER
		76 ;	
0031	C9	77	RET ; RETURN TO FORTRAN
		78 ;	
		79 ;*****	
		80 ;	
		81 ; SUBROUTINE AIOCRS	
		82 ;	
		83 ; AIOCRS IS THE CASSETTE OUTPUT DRIVER ROUTINE	
		84 ;	
		85 ; INPUTS: ADDRESS OF TEMP10 IN BC	
		86 ; OUTPUTS: DATA TO CASSETTE	
		87 ; CALLS: NOTHING	
		88 ; DESTROYS: A,F/F'S, B,C,D,E	
		89 ; MODIFIES: NOTHING	
		90 ; VERSION 1.0	
		91 ;	
		92 ; USE FROM FORTRAN CALL:	
		93 ;	
		94 ; CALL AIOCRS (TEMP10)	
		95 ;	
		96 ; TEMP10 = 16 BIT DATA WORD	
		97 ; TEMP10+1 = CONTROL FLAG BYTE	
		98 ; TEMP10+2 = ERROR FLAG BYTE	
		99 ;	
0000		100	CSCNTL EQU 0H ; NULL CONTROL BYTE
0001		101	COK EQU 1H ; CASSETTE OK STATUS
0000		102	CBEOT EQU 0H ; CASSETTE BEOT STATUS
0009		103	COUTH EQU 0AH ; MSB OUTPUT PORT
0000		104	COUTL EQU 02H ; LSB OUTPUT PORT
000B		105	CSTAT EQU 0AH ; STATUS PORT
000A		106	CCNTL EQU 0AH ; CONTROL PORT
0001		107	CWCNTL EQU 1H ; WRITE CONTROL BYTE
0002		108	CFCNTL EQU 2H ; FORWARD CONTROL BYTE
		109 ;	

LOC	OBJ	LINE	SOURCE STATEMENT
		110	; STATUS BYTE D2 = NOT BEOT, D1 - D7 SPARES
		111	RIOCAS:
		112	; BC POINTS TO TEMP10 ADDRESS
0032	0A	113	LDA B ; GET LSB OF DATA
0033	5F	114	MOV E, A ; PUT IT IN E
0034	C5	115	PUSH B ; SAVE BC ON STACK
0035	03	116	INX B ; INCREMENT BC TO MSB
0036	0A	117	LDA B ; GET MSB INTO ACC
0037	57	118	MOV D, A ; PUT IT IN D
		119	; DE HOLDS DATA (TEMP10)
0038	03	120	INX B ; POINT TO CONTROL BYTE
0039	0A	121	LDA B ; GET CONTROL BYTE
003A	07	122	INX B ; POINT BC TO ERROR FLAG BYTE
		123	;
		124	; TEST INPUT CONTROL BYTE FOR FUNCTION
003B	FE01	125	CPI 01H ; TEST IF CONTROL = 1
003D	CA4A00	C 126	JZ CWRITE ; IF EQUAL GO TO WRITE
0040	FE02	127	CPI 02H ; TEST IF CONTROL = 2
0042	CAEC00	C 128	JZ CFRWD ; IF EQUAL GO TO FORWARD
0045	3E02	129	MVI A, 02H ; SET ERROR = 2 FOR ILLEGAL COMMAND
0047	C38900	C 130	JMP CEXIT ; UNDEFINED OPERATION, EXIT
		131	;
		132	CWRITE:
004A	D80B	133	IN CSTAT ; READ STATUS WORD
004C	E601	134	RNI 01H ; KILL ALL BUT BEOT BIT
004E	FE01	135	CPI COK ; TEST IF STATUS OK
0050	C29300	C 136	JNZ CERROR ; IF ERROR GO TO PROCESS
0053	7B	137	MOV A, E ; GET LSB INTO ACC
0054	D300	138	OUT COUTL ; SEND BYTE TO PORT
0056	7A	139	MOV A, D ; GET MSB INTO ACC
0057	D309	140	OUT COUTH ; OUTPUT BYTE TO PORT
0059	3E01	141	MVI A, CMCNTL ; GET WRITE CONTROL BYTE
005B	D30A	142	OUT CCNTL ; SEND TO CONTROL PORT
		143	; HOLD START PULSE FOR 10 USEC (MIN)
005D	3E05	144	MVI A, 05H ; SET COUNT IN ACC
005F	3D	145	CLOOP: DCR A ; DECREMENT COUNTER
0060	C25F00	C 146	JNZ CLOOP ; IF NOT ZERO LOOP
0063	3E00	147	MVI A, CMCNTL ; SEND NULL CODE TO CASSETTE
0065	D30A	148	OUT CCNTL ; SEND TO CONTROL PORT
0067	3E00	149	MVI A, 0 ; LOAD ZERO FOR ERROR FLAG
0069	C38900	C 150	JMP CEXIT
		151	;
		152	CFRWD:
006C	D80B	153	IN CSTAT ; TEST STATUS
006E	E601	154	RNI 01H ; KEEP BEOT BIT
0070	FE00	155	CPI CBEOT ; TEST IF B OR E OF TAPE FLAG
0072	CA9300	C 156	JZ CERROR ; IF BEOT LOW ERROR
0075	3E02	157	MVI A, CFCNTL ; LOAD FFWD BYTE
0077	D30A	158	OUT CCNTL
0079	CD0000	E 159	CALL MSEC2H
007C	3E00	160	MVI A, 0
007E	D30A	161	OUT CCNTL
0080	C38900	C 162	JMP CEXIT
		163	;
474		164	CERROR:

ORIGINAL PAGE IS
OF POOR QUALITY

1515-II 0000-0095 MACRO ASSEMBLER, V3.0

A1010X PAGE 4

LOC	OBJ	LINE	SOURCE STATEMENT
0000	0000	155	MVI A,CCTRL ; STOP BYTE
0005	010A	166	OUT CCTRL
0007	1E01	167	MVI A,01; SET ERROR FLAG E001
		168	CEXIT:
0009	02	169	STAX B ; STORE ERROR BYTE IN MEMORY LOCATION (BC)
000A	C1	170	POP B ; SET BC BACK AS IN CALL
000B	C9	171	RET ; RETURN TO FORTRAN
		172	;
		173	END

PUBLIC SYMBOLS

A10C05 C 0032 A10C10 C 0000 A10V0T C 0010

EXTERNAL SYMBOLS

MSEC2H E 0000

USER SYMBOLS

A10C05 C 0032	A10C10 C 0000	A10V0T C 0010	CE00T A 0000	CCTRL A 000A	CEPR0R C 0003	CEXIT C 0009
CFCTRL A 0002	CFPR0R C 000C	CL00P C 005F	CXK A 0001	COU7L A 0009	COU7H A 0009	CCTRL A 0000
CSTAT A 000B	CWCTRL A 0001	CWRITE C 004A	MSEC2H E 0000			

ASSEMBLY COMPLETE. NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	
		2 NAME VCS100	
		3 ;	
		4 PUBLIC A10ITZ, A10ADC, ASETIT, A10CDU, CLKITZ, ACCITZ	
		5 ;	
		6 EXTRN MSEC2H ; CALL TO MONITOR WAIT 200 M SEC	
		7 ;	
		8 CSEG	
		9 ;	
		10 ;*****	
		11 ;	
		12 ; SUBROUTINE A10ADC (I,J)	
		13 ;	
		14 ; A10ADC IS THE A/D CONVERTER I/O DRIVER ROUTINE	
		15 ;	
		16 ; INPUTS: BC,DE ADDRESS POINTERS	
		17 ; OUTPUTS: DATA IN (DE)	
		18 ; CALLS: NOTHING	
		19 ; DESTROYS: A,F/R'S	
		20 ; MODIFIES: WORK	
		21 ; VERSION: 1.0	
		22 ;	
		23 A10ADC: ; (SINPNT, TEMP10)	
		24 ;	
		25 ; BC POINTS TO SINPNT - CHANNEL #, DE POINTS TO STORAGE LOCATION	
		25 ;	
0000	05	27	PUSH B ; SAVE BC ON STACK
0001	05	28	PUSH D ; SAVE DE ON STACK
0002	0A	29	LDAX B ; GET VALUE OF SINPNT INTO A
0003	2D	30	DCR A ; SET POINTER -1 TO 0-7 FROM 1-8
0004	F3	31	DI ; DISABLE INTERRUPTS
0005	260A	32	MVI H, 0AH ; SET UP TIME OUT COUNTER
0007	D360	33	OUT MPXOTR ; SEND CHAN SELECT TO MPX OUTPUT REGISTER
0009	002600	C 34	CALL WAIT50
		35	ADCG0:
000C	25	36	DCR H ; DECREMENT COUNT
000D	CA1700	C 37	JZ ADCG0 ; IF TIME OUT EXIT LOOP
0010	D681	38	IN MPXSI1 ; SAMPLE A/D DONE
0012	E680	39	ANI 80H ; TEST MSB
0014	CA0C00	C 40	JZ ADC05 ; IF NOT READY LOOP
0017	D694	41	ADCG0: IN MPXDA1 ; GET H.S. BYTE
0019	E60F	42	ANI 0FH ; KEEP 12 BITS
001B	47	43	MOV B, A ; SAVE IN B REG
001C	D695	44	IN MPXDA2 ; GET L.S. BYTE
001E	12	45	STAX D ; STORE AT TEMP10 LSB
001F	FB	46	EI ; REENABLE INTERRUPTS
0020	78	47	MOV A, B ; GET H.S. BYTE INTO A
0021	13	48	INX D ; SET POINTER TO MSB OF TEMP10
0022	12	49	STAX D ; STORE AT TEMP10 MSB
0023	01	50	POP D ; RESTORE DE
0024	01	51	POP B ; RESTORE BC
0025	09	52	RET ; RETURN TO FORTRAN
		53 ;	
476		54 ;	DELAY ROUTINE

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0 VCSIOD PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
0026	060A	55	WAIT50: MVI B,0AH ; LOAD COUNT FOR 50 USEC DELAY
0028	05	56	WAITDC: DCR B
0029	022900	57	JNZ WAITDC
002C	09	58	RET
		59	;
0084		60	MPXDA1 EQU 04H ; MPX DATA 1
0085		61	MPXDA2 EQU 05H ; MPX DATA 2 HEX 05 PORT
0086		62	MPXOTR EQU 00H ; MPX OUTPUT REGISTER HEX 00 PORT
0081		63	MPXSTP EQU 01H ; MPX STATUS INPUT REGISTER HEX 01 PORT
		64	;
		65	*****
		66	;
		67	SUBROUTINE AGETIT (L,J)
		68	;
		69	AGETIT TAKES THE HEX ADDRESS OF TEMP10 IN BC AND
		70	LOADS THE VALUE AND PUTS IT IN (DE)
		71	;
		72	INPUTS: BC,DE
		73	OUTPUTS: HEX VALUE IN J AT (DE)
		74	CALLS: NOTHING
		75	DESTROYS: A,F/F'S
		76	MODIFIES: TEMP10
		77	VERSION: 1.0
		78	;
		79	AGETIT:
002D	0A	80	LDAX B ; LOAD A WITH (BC)
002E	6F	81	MOV L,A ; STORE LSB IN L FOR MEM REF
002F	03	82	INX B ; INCREMENT BC
0030	0A	83	LDAX B ; LOAD A WITH (BC)
0031	67	84	MOV H,A ; STORE IN H MSB FOR POINT TO SINPHT
		85	HL IS ADDRESS TO GET FOR DISPLAY
0032	7E	86	MOV A,M ; LOAD A WITH DATA 1
0033	13	87	INX D ; POINT DE TO DOB(6) FOR FORMAT PACK REVERSE
0034	12	88	STAX D ; STORE DATA AT (DE)
0035	1B	89	DCX D ; DECREMENT DC TO DOB(5) FOR NEXT BYTE
0036	23	90	INX H ; INCREMENT HL
0037	7E	91	MOV A,M ; LOAD A WITH DATA 2
0038	12	92	STAX D ; STORE AT (DE)
0039	09	93	RET ; RETURN TO FORTRAN
		94	;
		95	*****
		96	;
		97	SUBROUTINE AIOITZ
		98	;
		99	AIOITZ INITIALIZES THE I/O INTERFACES
		100	;
		101	INPUTS: NONE
		102	OUTPUTS: NONE
		103	CALLS: NOTHING
		104	DESTROYS: A,F/F'S,BC,DE,HL
		105	MODIFIES: NOTHING
		106	VERSION: 1.0
		107	;
		108	AIOITZ:
003A	F3	109	DI ; DISABLE INTERRUPTS

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 8080/8085 MICRO ASSEMBLER, V3.0 VCS100 PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
		110 ;	
003B	3E00	111	MVI A,0 ; LOAD A REG WITH 0
003D	D307	112	OUT 07 ; PANEL
003F	D308	113	OUT 08 ; CASSETTE
0041	D309	114	OUT 09 ; CASSETTE
0043	D30A	115	OUT 0AH ; CASSETTE
0045	3EFF	116	MVI A,0FFH ; LOAD ALL ONES (NEGATIVE LOGIC)
0047	D303	117	OUT 03 ; APM VALVES
0049	3EF0	118	MVI A,0F0H ; SEND 00001111
004B	D306	119	OUT 06 ; PANEL
		120 ;	
		121 ;	CALL A/D INITIALIZE
004D	CD5A00	C 122	CALL ADCITZ
		123 ;	
		124 ;	CALL CLOCK INITIALIZE
0050	CD6F00	C 125	CALL CLKITZ
		126 ;	
		127 ;	SET UP NBS
0053	0B35	128	IN 035H ; SET NBS TO DIS COUNT
0055	0B37	129	IN 033H ; LATCH DATA
0057	0B32	130	IN 032H ; RESET SYSTEM
		131 ;	
0059	C9	132	RET ; RETURN
		133 ;	
		134 ;	*****
		135 ;	
		136 ;	SUBROUTINE ADCITZ
		137 ;	
		138 ;	ADCITZ INITIALIZES THE DATA TRANSLATION A/D
		139 ;	
		140	ADCITZ: ; ENTRY POINT
		141 ;	
005A	3E0F	142	MVI A,0FH ; LOAD PORT A,MODE 0
005C	D382	143	OUT MPXOCP ; MULTIPLEXOR OUTPUT CONTROL PORT
005E	3E07	144	MVI A,07H ; DISABLE INTERRUPT
0060	D382	145	OUT MPXOCP ; MUX OUT CONTROL PORT
0062	3E0F	146	MVI A,0CFH ; LOAD PORT B,MODE 3
0064	D383	147	OUT MPXSOP ; MULTIPLEXOR STATUS CONTROL PORT
0066	3EFF	148	MVI A,0FFH ; SET FOR INPUT BITS
0068	D383	149	OUT MPXSOP ;
006A	3E07	150	MVI A,07H ; DISABLE INTERRUPTS
006C	D382	151	OUT MPXSOP
006E	C9	152	RET ; RETURN
		153 ;	
0083		154	MPXSOP EQU 083H ; 83 HEX FOR MPX STAT CONTROL PORT
0082		155	MPXOCP EQU 082H ; 82 HEX FOR MPX OUT CONTROL PORT
		156 ;	
		157 ;	*****
		158 ;	
		159 ;	SUBROUTINE CLKITZ
		160 ;	
		161 ;	CLKITZ INITIALIZES THE TIMEF UNITS FOR NORMAL OPERATION
		162 ;	
0023		163	CLKONT EQU 023H ; CLOCK CONTROL PORT
		164 ;	

LOC	OBJ	LINE	SOURCE STATEMENT
006F	3E00	165	CLKITZ: MVI A,0 ; CLEAR ACC
0071	0325	166	OUT 25H ; STOP TIMER GATE 0,1,2
		167	; INITIALIZE TIMER 0 TO 1 MSEC
0073	3E34	168	MVI A,34H ; 00110100 CTL 0,L-M,MODE 2,EIN 34H
0075	0323	169	OUT CLKCNT ; CONTROL PORT
0077	3E71	170	MVI A,71H ; LSB OF 0271H
0079	0320	171	OUT 20H ; COUNTER PORT 0
007B	3E02	172	MVI A,02H ; MSB OF 0271H
007D	0320	173	OUT 20H ; COUNTER PORT 0
		174	; INITIALIZE TIMER 1 TO 100 MSEC
007F	3E74	175	MVI A,74H ; 01110100 CTL 1,L-M,MODE 2,PIN 74H
0081	0323	176	OUT CLKCNT ; CONTROL PORT
0083	3E64	177	MVI A,64H ; LSB OF 0064H
0085	0321	178	OUT 21H ; COUNTER PORT 1
0087	3E00	179	MVI A,00H ; MSB OF 0064H
0089	0321	180	OUT 21H ; COUNTER PORT 1
		181	; INITIALIZE TIMER 2 TO 1 SEC
008B	3E84	182	MVI A,84H ; 10110100 CTL 2,L-M,MODE 2,EIN 84H
008D	0323	183	OUT CLKCNT ; CONTROL PORT
008F	3EE8	184	MVI A,EE8H ; LSB OF 03EE8H
0091	0322	185	OUT 22H ; COUNTER PORT 2
0093	3E03	186	MVI A,03H ; MSB OF 03EE8H
0095	0322	187	OUT 22H ; COUNTER PORT 2
		188	; START COUNTERS
0097	3E07	189	MVI A,07H ; 0111, GATE 1,2,3
0099	0325	190	OUT 25H ; GATE PORT
009B	C9	191	RET ; RETURN
		192	;
		193	*****
		194	;
		195	; SUBROUTINE AIOCDU (BUF)
		196	;
		197	; AIOCDU READS DATA FROM THE C/D PANEL PORT 04
		198	;
		199	; INPUTS: BC POINT TO WORK BUFFER
		200	; OUTPUTS: NONE
		201	; CALLS: HSEC2H
		202	; DESTROYS: A,F/F'S,DE,HL
		203	; MODIFIES: /WORK/
		204	; VERSION 1.0
		205	;
		206	AIOCDU:
009C	16FF	207	MVI D,0FFH ; SET D AS FLAG TO FF
		208	; FRONT PANEL I/O DRIVER
		209	; CLEAR LATCHES, READ PORT AND CLEAR LATCHES
		210	; NOTE : LATCHES INVERT SWITCH DATA LINES TO BUS
		211	;
		212	CDURD: ; LOOP ENTRY POINT
009E	3E00	213	MVI A,00H ; LOAD 00000000 INTO A
00A0	0306	214	OUT CDUCLR ; SEND CLEAR PULSE
00A2	3E80	215	MVI A,80H ; TURN ON LATCH BIT ONLY
00A4	0306	216	OUT CDUCLR ; REMOVE CLEAR PULSE
		217	;
00A6	0E04	218	IN CDUINP ; READ C/D PORT
00A8	2F	219	CMA ; INVERT BITS FROM INTERFACE (NEG LOGIC)

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
00A9	E57F	220	ANI 7FH ; MASK NON EXISTANT BIT
		221	; TEST IF RUN MODE OR CALIBRATE
00AB	5F	222	MOV E,A ; SAVE INPUT IN E REG
00AC	E504	223	ANI 04H ; AND WITH 00000100 TO SEE RUN BIT
00AE	FE04	224	CPI 04H ; TEST IF ON
00B0	CACB00	C 225	JZ CDUGO ; TEST IF EQUAL (ON) AND JUMP TO GO
		226	; CALIBRATE MODE
00B3	7B	227	MOV A,E ; RESTORE INPUT
00B4	E60B	228	ANI 0EH ; 00001011 TEST NXT,RST,RCJ BITS FOR ON
00B6	FE0B	229	CPI 0EH ;
00B8	CAC100	C 230	JZ CCTEST ; IF = 0 SET ZERO FLAG SO JUMP TO TEST
		231	;
00BB	1600	232	MVI D,0 ; CLEAR FLAG IN D
00BD	63	233	MOV H,E ; SAVE NON ZERO INPUT IN H REG
00BE	C39E00	C 234	JMP CDURD ; GO BACK AND CONTINUE READ
		235	;
00C1	7A	236	CCTEST: MOV A,D ; TEST FLAG IN D
00C2	FE00	237	CPI 0
00C4	CAC000	C 238	JZ CDUGOX ; IF FLAG CLEARED GO TO UNPACK BITS
00C7	C39E00	C 239	JMP CDURD
		240	;
00CA	5C	241	CDUGOX: MOV E,H ; MOVE DATA TO E IF CAL MODE
		242	CDUGO:
00CB	7B	243	MOV A,E ; RESTORE INPUT BEFORE 0
00CC	F5	244	PUSH PSW ; SAVE A AND FLAGS
00CD	C5	245	PUSH B ; PUT ADDRESS IN HL
00CE	E1	246	POP H ; VIA STACK
00CF	16FF	247	MVI D,0FFH ; SET D TO 11111111 . TRUE
00D1	1E00	248	MVI E,0H ; SET E TO 00000000 . FALSE
00D3	0E08	249	MVI C,8 ; LOAD C WITH 8 COUNT
00D5	1F	250	CDU05: RAR ; ROTATE ACC INTO CARRY (RIGHT)
00D6	DADD77	C 251	JC CDUTRU ; IF BIT ON GO TO TRUE (CARRY ON (CMA) SEE NOTE)
00D9	73	252	MOV M,E ; STORE .FALSE. IN ARRAY(1)
00DA	C3DE00	C 253	JMP CDUDCR ; GO TO COUNT TEST
00DD	72	254	CDUTRU: MOV M,D ; STORE .TRUE. IN ARRAY(1)
00DE	00	255	CDUDCR: DCR C ; DECREMENT COUNTER C
00DF	C9E600	C 256	JZ CDUDON ; DONE ALL BITS
00E2	23	257	INX H ; HL = HL+1
00E3	C3D500	C 258	JMP CDU05 ; LOOP BACK FOR NEXT BIT
		259	CDUDON:
00E6	F1	260	POP PSW ; RECOVER INPUT BITS
00E7	E604	261	ANI 04H ; KEEP RUN BIT
00E9	FE04	262	CPI 04H ; TEST IF RUN MODE
00EB	CAF100	C 263	JZ CDUD10 ; SKIP WAIT IF RUN MODE
00EE	C00000	E 264	CALL HSEC2H ; WAIT FOR SWITCH DEBOUNCE
		265	CDUD10:
00F1	E5	266	PUSH H ; PUT ADDRESS BACK IN
00F2	C1	267	POP B ; BC VIA STACK
00F3	C9	268	RET ; RETURN TO FORTRAN
		269	;
006B		270	CLTEST EQU 06EH ; MASK FOR CALIBRATE TEST
0004		271	CDUINP EQU 04H ; PORT 4 FOR C/D INPUT
006F		272	CDTEST EQU 06FH ; MASK FOR RUN TEST
0006		273	CDUCLR EQU 06H ; PORT 6 FOR CDU LATCH CLEAR
		274	;

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 9090-9095 MACRO ASSEMBLER, V1.0

VC5100 PAGE 5

LOC	OBJ	LINE	SOURCE STATEMENT
			8 7 6 5 4 3 2 1
275	:		PORT 84 = X, PWR, MAN, AUT, PST, RUN/CAL, POS, INT
276	:		TYPE NC NC NO NC NC NC NC
277	:		VALUE X 1 1 0 1 1 1 1
278	:		NC - 0 ACTIVE WHEN PRESSED
280	:		END

PUBLIC SYMBOLS
AOCITZ C 005A AGETIT C 002D AIOFEC C 0000 AIOCDU C 009C AIOITZ C 003A CLKITZ C 006F

EXTERNAL SYMBOLS
MSEC2H E 0000

USER SYMBOLS	AOCGO C 0017	AOCITZ C 005A	AGETIT C 002D	AIOFEC C 0000	AIOCDU C 009C	AIOITZ C 003A
APC05 C 000C	COTEST A 006F	CDU05 C 0005	CDUCLP A 0005	CDU010 C 00F1	CDUDCR C 000E	CDUDEN C 00E5
CCTEST C 00C1	CDU00X C 00CA	CDUINP A 0004	CDURD C 000E	CDUTRU C 000C	CLKCNT A 0023	CLKITZ C 006F
CDU00 C 00CB	MPYDA1 A 0084	MPYDA2 A 0085	MPXOCP A 0082	MPXOTR A 0080	MPXSCP A 0023	MPXSIP A 0081
CLTEST A 006B	WAIT50 C 0025	WAITDC C 0029				
MSEC2H E 0000						

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	
		2 NAME INTRPT	
		3 ;	
		4 PUBLIC INTRPT, INTRDS, INTREN	
		5 ;	
		6 EXTRN OUTCAS	
		7 ;	
		8 ; CLOCK INTERRUPT SERVICE ROUTINE	
		9 ;	
		10 CSEG	
		11 ;	
		12 ; *****	
		13 ;	
		14 ; SUBROUTINE INTRPT	
		15 ;	
		16 ; INTRPT IS THE CLOCK INTERRUPT SERVICE ROUTINE	
		17 ;	
		18 ; INPUTS: NONE	
		19 ; OUTPUTS: NONE	
		20 ; CALLS: OUTCAS	
		21 ; DESTROYS: NOTHING	
		22 ; MODIFIES: CLOCK STATUS AND COUNTER BYTES	
		23 ; VERSION: 1.0	
		24 ;	
		25 INTRPT:	
0000	F5	26	PUSH PSW ; SAVE R/F/F'S
0001	0924	27	IN 024H ; READ CLOCK DATA FAST
0003	E5	28	PUSH H ; SAVE HL
0004	05	29	PUSH D ; SAVE DE
0005	C5	30	PUSH B ; SAVE BC
0006	F5	31	PUSH PSW ; SAVE INPUT FROM CLOCK
		32 ;	
		33 ; INTERRUPTS ARE DISABLED, READ CLOCK	
		34 ;	
		35 ; CLEAR BYTE FLAGS	
0007	3E00	36	MVI A,0 ; CLEAR ACC
0009	21C220	37	LXI H,20C2H ; POINT H TO FIRST FLAG
000C	77	38	MOV M,A ; CLEAR FLAG
000D	23	39	INX H
000E	77	40	MOV M,A ; CLEAR 100 FLAG
000F	23	41	INX H
0010	77	42	MOV M,A ; CLEAR SECOND FLAG
0011	F1	43	POP PSW ; GET INPUT BACK IN A REG
0012	2F	44	CMA ; INVERT BITS TO GET LOW DATA ACTIVE = 1
0013	E607	45	ANI 07H ; MASK UPPER 5 BITS
0015	0F	46	RRC ; SHIFT 1 SEC LINE INTO CARRY
0016	F5	47	PUSH PSW ; SAVE R/F/F'S
0017	0C5700	48	CC M1SEC ; TEST FOR BIT SET IN CARRY
001A	F1	49	POP PSW ; RESTORE A AFTER CALL
001B	0F	50	RRC ; SHIFT 100 MSEC BIT TO CARRY
001C	F5	51	PUSH PSW ; SAVE R/F/F'S
001D	0C4000	52	CC HUND0 ; TEST FOR BIT SET
0020	F1	53	POP PSW
0021	0F	54	RRC ; SHIFT 1 MSEC BIT TO CARRY

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
0022	003400	C	55 CC ONESEC ; TEST FOR BIT SET
			56 ;
			57 EXIT:
			58 ; WAIT FOR LINES TO GO HIGH AGAIN
0025	0024		59 IN 024H
0027	E607		60 ANI 07H
0029	FE07		61 CPI 07H
002B	C22500	C	62 JNZ EXIT
			63 ;
002E	C1		64 POP B ; RESTORE BC
002F	D1		65 POP D ; RESTORE DE
0030	E1		66 POP H ; RESTORE HL
0031	F1		67 POP PSW ; RESTORE A/F/F'S
			68 ;
0032	FB		69 EI ; ENABLE INTERRUPTS
0033	C9		70 RET ; RETURN TO MONITOR CALLING POINT
			71 ;
			72 ONESEC:
0034	21C420		73 LXI H,20C4H ; SET HL TO SECOND FLAG BYTE
0037	3EFF		74 MVI A,0FFH ; LOAD A WITH 11111111
0039	77		75 MOV M,A ; SET FLAG BYTE
003A	23		76 INX H ; POINT HL TO COUNT BYTE
003B	3E01		77 MVI A,01H ; SET A TO 1
003D	86		78 ADD M ; ADD COUNT TO ACC
003E	77		79 MOV M,A ; STORE COUNT BACK IN MEMORY BYTE
003F	23		80 INX H ; POINT TO MSB
0040	3E00		81 MVI A,00H ; ADD ZERO AND CARRY TO MSB COUNT
0042	86		82 ADD M ; ADD CARRY TO COUNT MSB
0043	77		83 MOV M,A ; SAVE IT AWAY
			84 ;
0044	3E00		85 MVI A,0
0046	D325		86 OUT 25H ; SET GATE TRIGGER LOW-HIGH TO SYNCPONIZE
0048	3E07		87 MVI A,07H
004A	D325		88 OUT 25H
004C	C9		89 RET
			90 ;
			91 HUNDRED:
004D	21C320		92 LXI H,20C3H ; SET HL TO 100 MSEC FLAG BYTE
0050	3EFF		93 MVI A,0FFH ; SET ACC TO 11111111
0052	77		94 MOV M,A ; SET FLAG BYTE
0053	C00000	E	95 CALL OUTCDS
0056	C9		96 RET
			97 ;
			98 MILLISEC:
0057	21C220		99 LXI H,20C2H ; HL POINTS TO 1 MSEC FLAG BYTE
005A	3EFF		100 MVI A,0FFH ; SET ACC TO 11111111
005C	77		101 MOV M,A ; SET FLAG BYTE TRUE
005D	C9		102 RET
			103 ;
			104 ; *****
			105 ;
			106 ; SUBROUTINE INTRODS
			107 ;
			108 ; INTRODS DISABLES THE CPU INTERRUPT LINES
			109 ;

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

INTPPT PAGE 2

LCC	OBJ	LINE	SOURCE STATEMENT
		110	; INPUTS: NONE
		111	; OUTPUTS: NONE
		112	; CALLS: NOTHING
		113	; DESTROYS: NOTHING
		114	; MODIFIES: INTERRUPT MASK FOR CPU
		115	; VERSION: 1.0
		116	;
		117	INTROS:
005E	F3	118	DI
005F	C9	119	RET ; RETURN AFTER DISABLING INTERRUPTS
		120	;
		121	;*****
		122	;
		123	SUBROUTINE INTREN
		124	;
		125	INTREN ENABLES THE CPU INTERRUPT LINES
		126	;
		127	; INPUTS: NONE
		128	; OUTPUTS: NONE
		129	; CALLS: NOTHING
		130	; DESTROYS: A,F/F'S
		131	; MODIFIES: CPU INTERRUPT MASK
		132	; VERSION: 1.0
		133	;
		134	INTREN:
0060	3E08	135	MVI A,08H ; LOAD A WITH INTERRUPT MASK
0062	30	136	SIM ; SET INTERRUPT MASK
0063	F8	137	EI ; ENABLE INTERRUPTS
0064	C9	138	RET ; RETURN
		139	;
		140	END

PUBLIC SYMBOLS

INTROS C 005E INTREN C 0060 INTRPT C 0000

EXTERNAL SYMBOLS

OUTCAS E 0000

USER SYMBOLS

EXIT C 0025 HUNDRO C 004D INTROS C 005E INTREN C 0060 INTRPT C 0000 NILEEC C 0057 ONESEC C 0034
OUTCAS E 0000

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	
		2 ; SUBROUTINE DEFAULT	
		3 ;	
		4 ; DEFAULT IS A TABLE OF OPERATION DEFAULT DATA IN ROM	
		5 ;	
		6 ;*****	
		7 ;	
		8 ; COMMON/DEFAULT/ DEFIOF,DEFCTV,DEFPRM,DEFPNT,ICMPNT,NMXVEC,NMXIPX	
		9 ;	
		10 ; DEFIOV(20) LOG*1	
		11 ; DEFPRM(20) INT*2	
		12 ; DEFCTV(16) LOG*1	
		13 ; DEFPNT(8) INT*2	
		14 ; ICMPNT(10) INT*2	
		15 ; NMXVEC(16) INT*2	
		16 ; NMXIPX(6) INT*1	
		17 ;	
		18 NAME DEFAULT	
		19 ;	
		20 PUBLIC DEFAULT	
		21 ;	
0850		22 ORG 0850H	
		23 ;	
		24 DEFAULT: ; PUBLIC SYMBOL ENTRY POINT	
0850	FF	25 DEFIOV: DB 0FFH ; IOFVEC(1) = SEN	
0851	FF	26 DB 0FFH ; EFD	
0852	FF	27 DB 0FFH ; PCF	
0853	FF	28 DB 0FFH ; LCF	
0854	FF	29 DB 0FFH ; LAP	
0855	FF	30 DB 0FFH ; RAP	
0856	FF	31 DB 0FFH ; CID	
0857	FF	32 DB 0FFH ; PKF	
0858	00	33 DB 000H ; PKR	
0859	FF	34 DB 0FFH ; VCF	
085A	FF	35 DB 0FFH ; VCR	
085B	FF	36 DB 0FFH ; VOT	
085C	FF	37 DB 0FFH ; DSP	
085D	00	38 DB 00H ; KEY	
085E	00	39 DB 00H ; CAS	
085F	00	40 DB 00H ; CRT	
0860	FF	41 DB 0FFH ; CDU	
0861	00	42 DB 00H ; DUMMY	
0862	00	43 DB 00H ; DUMMY	
0863	00	44 DB 00H ; DUMMY	
		45 ;	
		46 DEFPRM: ; /DEFPRM/ ; INT*2 PARAMETERS	
0864	0000	47 DW 0000H ; MAX DRUM VALUE DMAX	
0866	00FE	48 DW -200H ; MIN DRUM VALUE DMIN	
0869	0000	49 DW 9 ; ERROR LIMIT COUNT FOR TRIP MODE	
086A	F000	50 DW 0F0H ; CID STEP	
086C	0000	51 DW 0 ; PICK AVEFAGING NUMBER	
086E	F000	52 DW 0F0H ; PICK STEP	
0870	0000	53 DW 00H ; FRONT DEAD BAND	
0872	5000	54 DW 50H ; REAR DEAD BAND	

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 9000/8085 MACRO ASSEMBLER, V3.0

DEFAULT PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
0874	0100	55	DW 1 ; LENGTH OF EICKOFF ARMS
0876	3E00	56	DW 54 ; RADIUS OF DRUMS
0878	0A00	57	DW 10 ; LAST CUT POSITION
087A	1100	58	DW 17 ; PRESENT CUT POSITION
087C	0000	59	DW 0 ; CID POSITION
087E	6800	60	DW 68H ; FTRLOC
0880	0100	61	DW 1 ; DFDCT
0882	0100	62	DW 1 ; DSTP
0884	E000	63	DW 0E0H ; FRONT DBIAS
0886	0000	64	DW 0 ; REAR DRUM BIAS
0888	0000	65	DW 0 ; SRKVAL
088A	0000	66	DW 0 ; TRGFCT
		67 ;	
088C	FF	68	DEFTV: DB 0FFH ; LOG*1 CNMVEC(1) CID
088D	FF	69	DB 0FFH ; LCF
088E	00	70	DB 00H ; PKF
088F	00	71	DB 000H ; PKR
0890	FF	72	DB 0FFH ; PCF
0891	FF	73	DB 0FFH ; VCF
0892	FF	74	DB 0FFH ; VCR
0893	FF	75	DB 0FFH ; FRT
0894	FF	76	DB 0FFH ; RER
0895	FF	77	DB 0FFH ; CUT
0896	00	78	DB 0 ; DUMMY
0897	00	79	DB 0 ; DUMMY
0898	00	80	DB 0 ; DUMMY
0899	00	81	DB 0 ; DUMMY
089A	00	82	DB 0 ; DUMMY
089B	00	83	DB 0 ; DUMMY
		84 ;	
089C	0000	85	DEFPNT: DW 0 ; INT*2
089E	0000	86	DW 0
08A0	0000	87	DW 0
08A2	0000	88	DW 0
08A4	0000	89	DW 0
08A6	0000	90	DW 0
08A8	0000	91	DW 0
08AA	0000	92	DW 0
		93 ;	
		94 ; /OPPARM/	
		95 ;	
08AC	0700	96	ICNPNT: DW 7 ; INT*2 (1)
08AE	0400	97	DW 4
08B0	0800	98	DW 8
08B2	0900	99	DW 9
08B4	0300	100	DW 3
08B6	0900	101	DW 10
08B8	0800	102	DW 11
08BA	1200	103	DW 18
08BC	1300	104	DW 19
08BE	1400	105	DW 20
		106 ;	
08C0	0900	107	HMVVEC: DW 9 ; INT*2 SEN
08C2	0900	108	DW 9 ; ERR
08C4	0000	109	DW 0 ; SIR

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

DEFAULT PAGE 2

LOC	ORI	LINE	SOURCE STATEMENT
0B06	0700	110	DW 7 ; IPH
0B08	0800	111	DW 8
0B0A	0900	112	DW 9 ; DSP
0B0C	1400	113	DW 20 ; IOF
0B0E	0A00	114	DW 10 ; CHM CTY
0B10	0B00	115	DW 8 ; DBG
0B12	0900	116	DW 9 ; CBF
0B14	0800	117	DW 128 ; TAP
0B16	0800	118	DW 8 ; KIB
0B18	1400	119	DW 20 ; PRM
0B1A	0A00	120	DW 10 ; OPS
0B1C	0800	121	DW 8 ; PNT
0B1E	0800	122	DW 8 ; DUMMY
		123 ;	
0B20	11	124	MMXIPX: DB 17 ; MMXIPX(1) IOF VECTOR PATCHING
0B21	0A	125	DB 16 ; CONTROL VECTOR PATCHING
0B22	09	126	DB 9 ; OPERATION CONTROL PATCHING
0B23	14	127	DB 20 ; PARAMETERS TO BE RESET
0B24	01	128	DB 1 ; DATA INITIALIZE , CLEAR CMOS BUFFERS
0B25	20	129	DB 32 ; SENSOR TABLES
0B26	0A	130	DB 10 ; MOS CONTROL WORDS
		131 ;	
0C90		132	ORG 0C90H
		133 ;	
		134	DEFOPS:
		135	; COMMON /DEFOPS/ DEFOPS(10)
0C90	0200	136	DW 2 ; OFMODE
0C92	0100	137	DW 1 ; MODECT
0C94	0100	138	DW 1 ; RNGMSK
0C96	0100	139	DW 1 ; SENDEL
0C98	0000	140	DW 0 ; RATE CHECK
0C9A	0000	141	DW 0 ; STATUS
0C9C	0000	142	DW 13 ; FDVGN
0C9E	0002	143	DW 300H ; FDCAL
0CA0	0600	144	DW 6 ; RDVGN
0CA2	0003	145	DW 300H ; RDCAL
		146 ;	
0CA4		147	ORG 0CA4H
		148 ;	
		149	DEFMOS:
		150	; COMMON /DEFMOS/ DEFMOS(10)
0CA4	0060	151	DW 6000H ; MAXSLT
0CA6	0000	152	DW 0 ; STRIPT
0CA8	0000	153	DW 0 ; STEPSZ
0CAA	00A0	154	DW 0A000H ; NEGMAX
0CAC	0060	155	DW 6000H ; POSMAX
0CAE	0000	156	DW 0 ; LSTPNT
0CB0	0600	157	DW 6 ; ISNIND
0CB2	0000	158	DW 0 ; ISMPOS
0CB4	1001	159	DW 0110H ; CIDMAX
0CB6	FA00	160	DW 0FAH ; CIDMIN
		161 ;	
		162	END

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

DEFAULT PAGE 4

ORIGINAL PAGE IS
OF POOR QUALITY

PUBLIC SYMBOLS
DEFAULT A 0B50

EXTERNAL SYMBOLS

USER SYMBOLS

DEFAULT A 0B50	DEFTY A 0B8C	DEFIOY A 0B50	DEFMOS A 0C84	DEFOPS A 0C90	DEFENTY A 0B9C	DEFPRM A 0B64
ICMPNT A 0BAC	MMXIPX A 0BEB	MMXVEC A 0BC0				

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1	NAME TABLE
		2	;
		3	PUBLIC ASCTAB
		4	;
		5	*****
		6	;
		7	COMMON /MESTAB/
		8	;
		9	DATA INITIALIZATION OF ASCII TABLES
		10	;
0A00		11	ORG 0A00H
		12	/MESTAB/ STARTS AT 0A00 HEX
		13	;
		14	ASCTAB:
		15	ALLOCATION OF ERMES(20) CHARACTER*4 ARRAY
0A00	2453454E	16	DB '\$SEN'
0A04	24444644	17	DB '\$CFD'
0A08	24504346	18	DB '\$PCF'
0A0C	244C4346	19	DB '\$LCF'
0A10	244C4150	20	DB '\$LAP'
0A14	24524150	21	DB '\$RAP'
0A18	24434944	22	DB '\$CID'
0A1C	24504B46	23	DB '\$PKF'
0A20	24504B52	24	DB '\$PKR'
0A24	24564346	25	DB '\$VCF'
0A28	24564352	26	DB '\$VCR'
0A2C	24564F54	27	DB '\$VOT'
0A30	24445350	28	DB '\$DSP'
0A34	244B4559	29	DB '\$KEY'
0A38	24434153	30	DB '\$CRS'
0A3C	24435254	31	DB '\$CRT'
0A40	24434455	32	DB '\$COU'
0A44	24465254	33	DB '\$FRT'
0A48	24524552	34	DB '\$RER'
0A4C	24434F54	35	DB '\$COT'
		36	;
		37	ALLOCATION OF DEGMES(20) CHARACTER*4 ARRAY
		38	;
0A50	44425547	39	DB 'DEBUG'
0A54	4F55543F	40	DB 'OUT?'
0A58	494E502F	41	DB 'IMP?'
0A5C	4441543F	42	DB 'DAT?'
0A60	49545A3F	43	DB 'ITZ?'
0A64	5441503F	44	DB 'TAP?'
0A68	492F4F3F	45	DB 'I/O?'
0A6C	434C4B3F	46	DB 'CLK?'
0A70	40454D3F	47	DB 'MEM?'
0A74	5244593F	48	DB 'RDY?'
0A78	454E443F	49	DB 'END?'
0A7C	5359532D	50	DB 'SYS~'
0A80	504F5254	51	DB 'PORT'
0A84	434B4E4C	52	DB 'CHNL'
0A88	20555020	53	DB ' UP '
0A8C	444F574E	54	DB 'DOWN'

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0 TABLE PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
0A90	40414E2D	55	DB 'MAN-'
0A94	404F4445	56	DB 'MODE'
0A98	57524954	57	DB 'WRIT'
0A9C	4144521F	58	DB 'ADR?'
		59 ;	
		60 ;	ALLOCATION OR RUNMES(20) CHARACTER*4 ARRAY
		61 ;	
0AA0	3F432F4F	62	DB '?I/O'
0AA4	3F434E54	63	DB '?CIT'
0AA8	3F4F5053	64	DB '?OPS'
0AAC	3F565242	65	DB '?VRS'
0AB0	2A52554E	66	DB '*RUN-OK*'
0AB4	2D4F482A		
0AB8	2A444542	57	DB '*DEBUG *'
0ABE	5547292A		
0AC0	3F43414C	68	DB '?CHLRAT'
0AC4	42524154		
0AC8	2A204841	69	DB '* HALT *'
0ACC	4C54202A		
0AD0	4D415348	70	DB 'MASK'
0AD4	5041524D	71	DB 'PARN'
0AD8	45525220	72	DB 'ERR '
0ADC	25205354	73	DB '%-STOP-%'
0AE0	4F502D25		
0AE4	2A2D4558	74	DB '*-EXIT-*'
0AE8	49542D2A		
0AEC	20232320	75	DB ' ## '
		76 ;	
		77 ;	ALLOCATION OF DSPCODE(10) CHARACTER*1 ARRAY
		78 ;	
0AF0	24202326	79	DB '\$*4&? =/'
0AF4	25403F20		
0AF8	3D2F		
		80 ;	
		81 ;	/MESTB2/ STARTS AT 0AF0H
		82 ;	
		83 ;	ALLOCATION OF PARNES(20) CHARACTER*4
		84 ;	
0AFA	444D4158	85	DB 'DMAX' ; MAX DRUM VALUE
0AFE	444D494E	86	DB 'DMIN' ; MIN DRUM VALUE
0B02	45524C4D	87	DB 'ERLM' ; ERROR LIMIT
0B06	43445354	88	DB 'COST' ; CID STEP
0B0A	50415645	89	DB 'PAVE' ; NUMBER OF PICK SIGNALS TO AVERAGE
0B0E	504B5350	90	DB 'PKSP' ; PICK STEP
0B12	44424446	91	DB 'DBDF' ; FRONT DEAD BAND FOR CONTROLLER
0B16	44424452	92	DB 'DBDR' ; REAR DEAD BAND
0B1A	4C454E47	93	DB 'LENG' ; LENGTH OF ARM
0B1E	44524144	94	DB 'DRAD' ; DRUM RADIUS
0B22	4C435053	95	DB 'LCPS' ; LAST CUT POSITION
0B26	50435053	96	DB 'PCPS' ; PRESENT CUT POSITION
0B2A	43445053	97	DB 'CDPS' ; CID POSITION
0B2E	464C4F43	98	DB 'FLOC' ; FRONT TO REAR DISTANCE
0B32	44464354	99	DB 'DFCT' ; DFD FACTOR
0B36	44535450	100	DB 'DSTP' ; DFD STEP UPDATE FOR CMOS
0B3A	46424153	101	DB 'FRAS' ; FRONT DRUM BIAS

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
0B3E	52404152	102	DB 'R2AS' ; REAR DRUM BIAS
0B42	53485256	103	DB 'SKRV' ; RATE CHECK PARAMETER
0B46	54524746	104	DB 'TRGF' ; TRIG FACTOR
		105	;
0C48		106	ORG 0C48H
		107	;
		108	;/MESTB3/ STARTS AT 0C48H
		109	;
		110	;/ ALLOCATION OF OPSMES(10) CHARACTER*4
		111	;
0C48	404F4445	112	DB 'MODE' ; MODE MODE
0C44	52414E47	113	DB 'RANG' ; RANGE CHECK
0C48	5344454C	114	DB 'SDEL' ; OPERATION SENSOR DELAY FLAG
0C4C	52415445	115	DB 'RATE' ; SENSOR RATE TEST
0C50	53545553	116	DB 'STUS' ; STATUS WORD
0C54	46444567	117	DB 'FDVG' ; FRONT DRUM GAIN
0C58	4644474C	118	DB 'FDCL' ; FRONT DRUM CAL
0C5C	52445E47	119	DB 'RDVG' ; REAR DRUM GAIN
0C60	5244434C	120	DB 'RDCL' ; REAR DRUM CAL
0C64	20202020	121	DB ' ' ;
		122	;
0CC8		123	ORG 0CC8H
		124	;
		125	;/MOSMES/ STARTS AT 0CC8H
		126	;
		127	;/ ALLOCATION OF MOSMES(16) CHARACTER*4
		128	;
0CC8	40585354	129	DB 'MXST'
0CCC	53545054	130	DB 'STPT'
0CD0	5350535A	131	DB 'SPSZ' ; STEP SIZE
0CD4	4E474D58	132	DB 'NGMX' ; NEGMAX
0CD8	50534D58	133	DB 'PSMX' ; POSMAX
0CDC	4C535054	134	DB 'LSPT' ; LSTPNT
0CE0	534E4944	135	DB 'SNID' ; ISNIND
0CE4	534E5053	136	DB 'SNPS' ; ISNPOS
0CE8	43444D58	137	DB 'COMX' ; CIDMAX
0CEC	43444D4E	138	DB 'CDMN' ; CIDMIN
0CF0	20202020	139	DB ' ' ;
0CF4	20202020	140	DB ' ' ;
0CF8	20202020	141	DB ' ' ;
0CFC	20202020	142	DB ' ' ;
0D00	20202020	143	DB ' ' ;
0D04	20202020	144	DB ' ' ;
		145	;
		146	END

PUBLIC SYMBOLS
ASCTAB A 0A00

EXTERNAL SYMBOLS

USER SYMBOLS
ASCTAB A 0A00

ISIS-11 8080/8085 MACRO ASSEMBLER, V3.0

TABLE PAGE 4

ORIGINAL PAGE IS
OF POOR QUALITY

ASSEMBLY COMPLETE. NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
		1 ;	
		2 ;	SUBROUTINE DEFTAB
		3 ;	
		4 ;	
		5 ;	DEFAULT TABLE FOR CONVERSION AND RANGE DATA
		6 ;	
		7 ;	*****
		8 ;	
		9	NAME DEFTAB
		10 ;	
		11	PUBLIC DEFTBL
		12 ;	
00E9		13	ORG 00E8H ; ORIGIN FOR COMMON
		14 ;	
		15 ;	COMMON/DEFTAB/ DEFRNG,DEFCON
		16 ;	
		17 ;	DEFRNG(10,2) INT*2
		18 ;	DEFCON(10,2) INT*2
		19 ;	
		20 ;	RANGE CHECK TABLE
		21 ;	
		22	DEFTBL: ;
		23 ;	MINIMUMS
		24	DEFRNG: ; RANGE DEFAULTS
00E9	00F0	25	DW -1000H ; DFD
00EA	00F0	26	DW -1000H ; PCF
00EC	00F0	27	DW -1000H ; LCF
00EE	00F0	28	DW -1000H ; LAP
00F0	00F0	29	DW -1000H ; RRP
00F2	0000	30	DW 00000 ; HBS CID MIN = 0
00F4	0000	31	DW -32760 ;
00F6	0000	32	DW -32760 ;
00F8	0000	33	DW -32760 ;
00FA	0000	34	DW -32760 ;
		35 ;	MAXIMUMS
00FC	F07F	36	DW 32760 ; DFD
00FE	F07F	37	DW 32760 ; PCF
0000	F07F	38	DW 32760 ; LCF
0002	0010	39	DW 1000H ; LAP
0004	0010	40	DW 1000H ; RRP
0006	1027	41	DW 10000 ; HBS CID MAX
0008	F07F	42	DW 32760 ;
000A	F07F	43	DW 32760 ;
000C	F07F	44	DW 32760 ;
000E	F07F	45	DW 32760 ;
		46 ;	
		47 ;	CONVERSION TABLE
		48 ;	
		49 ;	OFFSET TERMS
0010	97FB	50	DEFCON: DW 0FB97H ; DFD
0012	6006	51	DW 0660H ; PCF
0014	6006	52	DW 0660H ; LCF
0016	00F0	53	DW 0F000H ; LAP
0018	00F0	54	DW 0F000H ; RRP

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8800/8885 MACRO ASSEMBLER, V3.0

DEFTAB PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
0C1A	0000	55	DW 0 ; NBS CID
0C1C	0000	56	DW 0 ;
0C1E	0000	57	DW 0 ;
0C20	0000	58	DW 0 ;
0C22	0000	59	DW 0 ;
		60	; GAIN-TERMS
0C24	0100	61	DW 1 ; DFD
0C26	FFFF	62	DW -1 ; PCF
0C28	FFFF	63	DW -1 ; LCF
0C2A	0400	64	DW 4 ; LAP
0C2C	0400	65	DW 4 ; RAP
0C2E	0100	66	DW 1 ; CID
0C30	0100	67	DW 1 ;
0C32	0100	68	DW 1 ;
0C34	0000	69	DW 0 ;
0C36	0000	70	DW 0 ;
		71	;
		72	END

PUBLIC SYMBOLS
DEFTBL A 00E0

EXTERNAL SYMBOLS

USER SYMBOLS
DEFCON A 0C10 DEFRA0 A 00E0 DEFTBL A 00E0

ASSEMBLY COMPLETE NO ERRORS

1515-11 8080/8085 MACRO ASSEMBLER V3.0 MONTR PAGE 4

LOC OBJ LINE SOURCE STATEMENT

```

1 ; *****
2 ;
3 ;          PROGRAM: STOMON MONITOR VER 1.0
4 ;
5 ;          ORIGINAL SOURCE STD-85, SEC-80, LG MONITORS, FRO-LOG
6 ;          7303 KEYBOARD MONITOR.
7 ;
8 ;          MODIFIED BY FOSTER-HILLER ASSOCS
9 ;          NOVEMBER 01, 1989
10 ;          FOR FRO-LOG STD BUS 8085 WITH 7303 AND LEFT
11 ;          VCS HARDWARE CONFIGURATION
12 ;
13 ; *****
14 ;
15 ; ABSTRACT
16 ; =====
17 ;
18 ; THIS PROGRAM IS A 2K MONITOR FOR THE INTEL 8085 AND
19 ; PROVIDES A MINIMUM LEVEL OF UTILITY FUNCTIONS FOR THE USER EMPLOYING
20 ; EITHER AN INTER-ACTIVE CONSOLE (I.E. TELETYPE) OR THE USER EMPLOYING
21 ; THE KEYBOARD-DISPLAY. THE KEYBOARD MONITOR ALLOWS THE USER TO PERFORM
22 ; SUCH FUNCTIONS AS MEMORY AND REGISTER MANIPULATION, PROGRAM LOADING,
23 ; PROGRAM EXECUTION, INTERRUPTION OF AN EXECUTING PROGRAM, AND
24 ; SYSTEM RESET. THE MONITOR ALSO PROVIDES MINIMAL TAPE I/O.
25 ;
26 ; PROGRAM ORGANIZATION
27 ; =====
28 ;
29 ; THE PROGRAM IS ORGANIZED AS FOLLOWS :-
30 ;     1) COLD START ROUTINE (RESET)
31 ;     2) WARM START - REGISTER SAVE ROUTINE
32 ;     3) INTERRUPT VECTORS
33 ;     4) KEYBOARD MONITOR
34 ;     5) TTY MONITOR
35 ;     6) LAYOUT OF RAM USAGE
36 ;
37 ; THE KEYBOARD MONITOR BEGINS WITH THE COMMAND RECOGNIZER, FOLLOWED BY
38 ; THE COMMAND ROUTINE SECTION, UTILITY ROUTINE SECTION AND MONITOR
39 ; TABLES. THE COMMAND AND UTILITY ROUTINES ARE IN ALPHABETICAL ORDER
40 ; WITHIN THEIR RESPECTIVE SECTIONS.
41 ; THROUGHOUT THE KEYBOARD MONITOR, A COMMENT FIELD BEGINNING
42 ; WITH "ARG - " INDICATES A STATEMENT WHICH LOADS A VALUE INTO
43 ; A REGISTER AS AN ARGUMENT FOR A FUNCTION. WHEN THE DESIRED VALUE
44 ; LIST OF KEYBOARD MONITOR ROUTINES
45 ; =====
46 ;
47 ; CMDND
48 ; -----
49 ; EXAM
50 ; GOCHD
51 ; SSTEP
52 ; SUBST
53 ; -----
54 ; CLEAR

```

LOC	OBJ	LINE	SOURCE STATEMENT
		55 ;	CLODS
		56 ;	CLOST
		57 ;	DISPC
		58 ;	ERR
		59 ;	GTHX
		60 ;	HWDSP
		61 ;	ININT
		62 ;	INSDG
		63 ;	MXTRG
		64 ;	OUTPT
		65 ;	RDKBD
		66 ;	RETF
		67 ;	RETT
		68 ;	RGLOC
		69 ;	RSTOR
		70 ;	SETRG
		71 ;	UFADJ
		72 ;	UPCOT
		73 ;	
		74	NAME MONITR
		75 ;	
		76 ;	*****
		77 ;	
		78 ;	SET CONDITIONAL ASSEMBLY FLAG
		79 ;	
		80 ;	*****
		81 ;	
		82 ;	
		83 ;	WAITS SET 0 ; 0=NO WAIT STATES
		84	; 1=A WAIT STATE IS GENERATED FOR EVERY M CYCLE
		85	; THE APPROPRIATE DELAY TIME MUST BE USED FOR
		86	; TTY DELAY OR SET UP SINGLE
		87	; STEP TIMER FOR EACH CASE
		88 ;	
		89 ;	
		90 ;	*****
		91 ;	
		92 ;	MONITOR EQUATES
		93 ;	
		94 ;	*****
		95 ;	
2000		96	RAMST EQU 2000H ; START ADDRESS OF RAM - THIS PROGRAM ASSUMES
		97 ;	THAT 256 BYTES OF RANDOM ACCESS MEMORY BEGIN AT THIS ADDRESS.
		98 ;	THE PROGRAM USES STORAGE AT THE END OF THIS SPACE FOR VARIABLES,
		99 ;	SAVING REGISTERS AND THE PROGRAM STACK.
		100 ;	
0017		101	RNUSE EQU 23 ; RAM USAGE - CURRENTLY 23 BYTES ARE USED FOR
		102	; /SAVING REGISTERS AND VARIABLES
		103 ;	
0012		104	SKLN EQU 10 ; MONITOR STACK USAGE - MAX OF 9 LEVELS
		105 ;	
000F		106	UBRLN EQU 15 ; 5 USER BRANCHES - 3 BYTES EACH
		107 ;	
0000		108	ADFLD EQU 0 ; INDICATES USE OF ADDRESS FIELD OF DISPLAY
0050	496	109	ADISP EQU 30H ; CONTROL CHARACTER TO INDICATE OUTPUT TO

ORIGINAL PAGE IS
OF POOR QUALITY.

1513-11 5050/5085 MACRO RESEMBLER, V3.0

MONITR PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
		110	; /ADDRESS FIELD OF DISPLAY
0001		111	CNTRL EQU 001H ; ADDRESS FOR SENDING CONTROL CHARACTERS TO
		112	; /DISPLAY CHIP
0011		113	COMMA EQU 11H ; COMMA FROM KEYBOARD
0000		114	CSNIT EQU 0 ; INITIAL VALUE FOR COMMAND STATUS REGISTER
0020		115	CSR EQU 20H ; OUTPUT PORT FOR COMMAND STATUS REGISTER
0054		116	DDISP EQU 94H ; CONTROL CHARACTER TO INDICATE OUTPUT TO
		117	; /DATA FIELD OF DISPLAY
0001		118	DOT EQU 1 ; INDICATOR FOR DOT IN DISPLAY
0000		119	DSPLY EQU 000H ; ADDRESS FOR SENDING CHARACTERS TO DISPLAY
0001		120	DTFLD EQU 1 ; INDICATES USE OF DATA FIELD OF DISPLAY
0000		121	EMPTY EQU 00H ; HIGH ORDER 1 INDICATES EMPTY INPUT BUFFER
20E9		122	MNSTK EQU RANST + 256 - RMUSE ; START OF MONITOR STACK
0000		123	NODOT EQU 0 ; INDICATOR FOR NO DOT IN DISPLAY
		124	; NMNC - DEFINED LATER ; NUMBER OF COMMANDS
		125	; NUMRG - DEFINED LATER ; NUMBER OF REGISTER SAVE LOCATIONS
0010		126	PERIO EQU 10H ; PERIOD FROM KEYBOARD
00D0		127	PMPRT EQU 0ADH ; PROMPT CHARACTER FOR DISPLAY (DASH)
000E		128	UNMSK EQU 0EH ; UNMASK INPUT INTERRUPT
20C8		129	USRER EQU RANST + 256 - (RMUSE + SKLN + USPLN) ; START OF USER
		130	; /BRANCH LOCATIONS
		131	;
		132	; ***** PUBLIC SYMBOLS FOR PROGRAM LINKAGE
		133	;
		134	PUBLIC CO, CI, RKBD, DISP1, CNVBN, HEXASC, MSEC1H
		135	PUBLIC DELAY, SECDT, MSEC2H, MSEC20, OUTPT, GO, MONITR
		136	;
		137	EXTRN INTPT
		138	;
		139	*****
		140	;
		141	;
		142	;
		143	*****
		144	;
		145	TRUE MACRO WHERE ; BRANCH IF FUNCTION RETURNS TRUE
		146	JC WHERE
		147	ENDM
		148	;
		149	FALSE MACRO WHERE ; BRANCH IF FUNCTION RETURNS FALSE
		150	JNC WHERE
		151	ENDM
		152	;
		153	;
		154	*****
		155	;
		156	;
		157	;
		158	*****
		159	;
		160	THE USART IS ASSUMED TO COME UP IN THE RESET POSITION (THIS
		161	FUNCTION IS TAKEN CARE OF BY THE HARDWARE). THE USART WILL
		162	BE INITIALIZED IN THE SAME WAY FOR EITHER TTY OR CRT
		163	INTERFACE. THE FOLLOWING PARAMETERS ARE USED:
		164	;

```

LOC 0000      LINE      SOURCE STATEMENT
              165 :      MODE INSTRUCTION
              166 :      =====
              167 :
              168 :      2 STOP BITS
              169 :      PARITY DISABLED
              170 :      8 BIT CHARACTERS
              171 :      BAUD RATE FACTOR OF 64
              172 :
              173 :      COMMAND INSTRUCTION
              174 :      =====
              175 :
              176 :      NO HUNT MODE
              177 :      NOT(RTS) FORCED TO 0
              178 :      RECIEVE ENABLED
              179 :      TRANSMIT ENABLED
              180 :
              181 :      *****
              182 :
              183 :      PRO-LOG KEYBOARD LAYOUT
              184 :
              185 :      14 - TTY  15 - SNGL 16 - **** 17 - EXAM  RESET
              186 :      MON      STEP      PEGS  SYSTEM
              187 :      -----
              188 :
              189 :      C      D      E      F-FLAG  I 12 - SUSS
              190 :
              191 :
              192 :      8 - H   9 - L   A      B      I 12 - GO
              193 :
              194 :
              195 :      4 - SPH 5 - SPL 6 - PCH 7 - PCL I 11 - , - NEXT
              196 :
              197 :
              198 :      0      1      2      3      I 10 - . - EXEC
              199 :
              200 :
              201 :      *****
              202 :
              203 :
              204 :      ***** "RESET" KEY ENTRY POINT - COLD START
              205 :      ***** RST 0 ENTRY POINT
              206 :
0000          207 ORG 0
              208 MONTR: ; MONITOR ENTRY POINT
0000 0000      209 INITUT: MVI A,0 ; CLEAR ACCUMULATOR
              210 :
              211 :      *****
              212 :
0002 000001    213 JMP CLOST ; GO FINISH COLD START
              214 CLOSK: ; THEN JUMP BACK HERE
0005 000F00    215 JMP RES10 ; SKIP OVER WARM START CODE
              216 :
              217 :      ***** RST 1 ENTRY POINT - WARM START
              218 :
0008          219 ORG 8

```

LOC	OBJ	LINE	SOURCE STATEMENT
		220 :	SAVE REGISTERS
0009	22E710	221	SHLD LSRV ; SAVE H & L REGISTERS
000B	E1	222	POP H ; GET USER PROGRAM COUNTER FROM TOP OF STACK
000C	22F210	223	SHLD PSRV ; /AND SAVE IT
000F	F5	224	PUSH PSW
		225 :	***** 010H IS RST 2 ENTRY POINT (NOT USED)
0010	E1	226	POP H ; LOAD R/F/F'S INTO HL
0014	22ED10	227	SHLD FSRV ; SAVE FLIP, FLOPS & REGISTER A
0014	210000	228	LXI H, 0 ; CLEAR H & L
0017	39	229	DAD SP ; GET USER STACK POINTER
		230 :	***** 018H IS RST 3 ENTRY POINT (NOT USED)
0019	22F420	231	SHLD SSRV ; /AND SAVE IT
001B	21ED10	232	LXI H, SSRV+1 ; SET STACK POINTER FOR SAVING
001E	F9	233	SPHL ; /REMAINING REGISTERS
001F	05	234	PUSH B ; SAVE B & C
		235 :	***** 020H IS RST 4 ENTRY POINT (NOT USED)
0019	15	236	PUSH D ; SAVE D & E
0021	02EF00	237	JMP RESET0 ; LEAVE ROOM FOR VECTORED INTERRUPTS
		238 :	
		239 :	***** TRAP ENTRY POINT *****
		240 :	
		241 :	***** TRAP ENTRY POINT
		242 :	
0024		243	ORG 24H
0024	039107	244	JMP TRAPPR
		245 :	
		246 :	***** RST 5 ENTRY POINT
		247 :	
0028		248	ORG 28H
0028	039207	249	JMP RESET5 ; RST 5 JMP VECTOR
		250 :	
		251 :	***** RST 5.5 ENTRY POINT
		252 :	
002C		253	ORG 2CH
002C	013000	254	CALL INTPT ; CALL SERVICE ROUTINE
002F	09	255	RET ; RETURN TO CODE WHERE INTERRUPT HAPPENED
		256 :	
		257 :	***** RST 6 ENTRY POINT
		258 :	
0020		259	ORG 20H
0020	039107	260	JMP RESET6 ; RST 6 JMP VECTOR
		261 :	
		262 :	***** RST 6.5 ENTRY POINT
		263 :	
0024		264	ORG 24H
0024	039107	265	JMP RESET65 ; RST 6.5 JMP VECTOR
		266 :	
		267 :	***** RST 7 ENTRY POINT
		268 :	
0028		269	ORG 28H
0028	039107	270	JMP RESET7 ; RST 7 JMP VECTOR
		271 :	
		272 :	***** RST 7.5 ENTRY POINT
002C		273	ORG 2CH
002C	039107	274	JMP RESET75 ; RST 7.5 JMP LOCATION

ORIGINAL PAGE IS
OF POOR QUALITY

1313-11 8090/8095 MACRO ASSEMBLER, V2.0

MONITOR PAGE 6

LOC	ORG	LINE	SOURCE STATEMENT
		275 ;	
		276 ;	*****
		277 ;	
		278 RES10 ;	CONTINUE SAVING USER STATUS
003F	20	279	RIM ; GET USER INTERRUPT STATUS AND INTERRUPT MASK
0040	E50F	280	ANI 0FH ; KEEP STATUS AND MASK BITS
0042	32F120	281	STA ISAV ; SAVE INTERRUPT STATUS AND MASK
0045	3E0E	282	MVI A,UNMSK ; UNMASK INTERRUPTS FOR MONITOR USE
0047	30	283	SIM
0048	F3	284	DI ; INTERRUPTS DISABLED WHILE MONITOR IS RUNNING
		285	; (EXCEPT WHEN WAITING FOR INTERRUPT)
		286 ;	CHECK FOR 7303 INSTALLATION
0049	D800	287	IN 000H ; READ S1,S2
004B	E6C0	288	ANI 0C0H ; KEEP UPPER TWO BITS
004D	47	289	MOV B,A ; SAVE BITS
004E	E580	290	ANI 080H
0050	C20040	291	JNZ 4000H ; IF 8 BIT ON GO TO PROGRAM
0053	78	292	MOV A,B ; GET BITS BACK
0054	E640	293	ANI 040H
0056	C20010	294	JNZ 1000H ; IF BIT 7 ON GO TO PPOLG MONITOR AT 1000H
		295 ;	
0059	C09703	296	CALL MSEC2H ; CALL 200 MSEC DELAY
		297 ;	
		298 ;	*****
		299 ;	
		300 ;	BEGINNING OF KEYBOARD MONITOR CODE
		301 ;	
		302 ;	*****
		303 ;	
005C	C0C001	304	CALL CLEAR
005F	3E50	305	MVI A,00H ; LOAD BUFFER EMPTY FLAG
0061	32FE20	306	STA IBUFF ; SET INPUT BUFFER EMPTY FLAG
		307 ;	
		308 ;	*****
		309 ;	
		310 ;	FUNCTION: CMND - COMMAND RECOGNIZER
		311 ;	INPUTS: NONE
		312 ;	OUTPUTS: NONE
		313 ;	CALLS: RDKBD,ERR,SUBST,EXAM,CATCHD,STEP
		314 ;	DESTROYS: A,B,C,D,E,H,L,F/F'S
		315 ;	
		316	CMND:
0064	21E920	317	LXI H,MISTK ; INITIALIZE MONITOR STACK POINTER
0067	F9	318	SPHL ; HL TO STACK POINTER STORE
0068	21F303	319	LXI H,RDYTB1 ; LOAD ADDRESS OF READY MESSAGE TO HL
006B	0E00	320	MVI C,8 ; LOAD C WITH CHARACTER COUNT
006D	C09E02	321	CALL OUTPT ; DISPLAY MESSAGE
		322 ;	READ KEYBOARD FUNCTION KEYS
0070	3E9F	323	MVI A,0FH ; 00001111
0072	03D0	324	OUT 000H ; OUTPUT TO D0
0074	0E00	325	MOKEY: IN 000H ; INPUT FROM D0
0076	E630	326	ANI 030H ; AND IN A 0011000
0078	C97400	327	JZ MOKEY ; LOOP BACK IF NO KEY PRESSED
007F	C0C001	328	CALL CLEAR ; CLEAR DISPLAY
007E	C0C002	329	MONRDX: CALL RDKBD ; READ KEYBOARD

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	CDS	LINE	SOURCE STATEMENT
0081	010500	330	LXI B,NUMC ; COUNTER FOR NUMBER OF COMMANDS IN C
0084	01E500	331	LXI H,CMDTB ; GET ADDRESS OF COMMAND TABLE
		332	CMD10:
0087	EE	333	CMP M ; RECOGNIZE COMMAND ?
0089	CA9500	334	JZ CMD15 ; YES - GO PROCESS IT
008B	23	335	INX H ; NO - NEXT COMMAND TABLE ENTRY
008C	00	336	DCR C ; END OF TABLE
00ED	C28700	337	JNZ CMD10 ; NO - GO CHECK NEXT ENTRY
		338	; YES - COMMAND UNKNOWN
0090	03FF01	339	JMP ERR ; DISPLAY ERROR MESSAGE AND GET ANOTHER COMMAND
		340	CMD15:
0093	21BA03	341	LXI H,CMDAD ; GET ADDRESS OF COMMAND ADDRESS TABLE
0096	00	342	DCR C ; ADJUST COMMAND COUNTER
		343	; COUNTER ACTS AS POINTER TO COMMAND ADDRESS TABLE
0097	09	344	DAD B ; ADD POINTER TO TABLE ADDRESS TWICE BECAUSE
0098	09	345	DAD B ; TABLE HAS 2 BYTE ENTRIES
0099	7E	346	MOV A,M ; GET LOW ORDER BYTE OF COMMAND ADDRESS
009A	23	347	INX H
009B	56	348	MOV H,M ; GET HIGH ORDER BYTE OF COMMAND ADDRESS IN H
009C	5F	349	MOV L,A ; PUT LOW ORDER BYTE IN L
		350	; COMMAND ROUTINE ADDRESS IS NOW IN H & L
009D	E9	351	PCHL ; BRANCH TO ADDRESS IN H & L BY LOADING PC
		352 ;	
		353 ; *****	
		354 ;	
		355 ; COMMAND ROUTINES	
		356 ;	
		357 ; *****	
		358 ;	
		359 ; FUNCTION: EXAM - EXAMINE AND MODIFY REGISTERS	
		360 ; INPUTS: NONE	
		361 ; OUTPUTS: NONE	
		362 ; CALLS: CLEAR, SETRG, EPR, RGNAM, RGLOC, UPDAT, GTHEY, NXPQ	
		363 ; DESTROYS: A,B,C,D,E,H,L,F/F'S	
		364 ;	
		365 EXAM:	
009E	000001	366	CALL CLEAR ; CLEAR DISPLAY
00A1	21EB03	367	LXI H,XMSG ; DISPLAY EXAMINE MESSAGE
00A4	0E04	368	MVI C,4
00A6	CD9E02	369	CALL OUTPT
00A9	CD5403	370	CALL SETRG ; GET REGISTER DESIGNATOR FROM KEYBOARD AND
		371	; /SET REGISTER POINTER ACCORDINGLY
		372	; WAS CHARACTER A REGISTER DESIGNATOR ?
		373	FALSE ERR : NO - DISPLAY ERROR MSG. AND TERMINATE COMMAND
00AC	D2FF01	374+	JNC EPR
		375	EXM05:
00AF	001A03	376	CALL RGNAM ; OUTPUT REGISTER NAME TO ADDRESS FIELD
00B2	CD0003	377	CALL RGLOC ; GET REGISTER SAVE LOCATION IN H & L
00B5	7E	378	MOV A,M ; GET REGISTER CONTENTS
00B6	32F800	379	STA CURDT ; STORE REGISTER CONTENTS AT CURRENT DATA
00B9	CD7003	380	CALL UPDAT ; UPDATE DATA FIELD OF DISPLAY
00BC	0601	381	MVI B,DTFLD ; ARG - USE DATA FIELD OF DISPLAY
00BE	CD0002	382	CALL GTHEY ; GET HEX DIGITS - WERE ANY DIGITS RECEIVED?
		383	FALSE EXML0 : NO - DO NOT UPDATE REGISTER CONTENTS
00C1	02C000	384+	JNC EXML0

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 6080/3085 MACRO ASSEMBLER V3.0

MONTR PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
0004	CD0003	385	CALL RGLOC ; YES - GET REGISTER SAVE LOCATION IN H & L
0007	73	386	MOV R6E ; UPDATE REGISTER CONTENTS
		387	EXM10:
0008	FE10	388	CPI PERIO ; WAS LAST CHARACTER A PERIOD ?
000A	CD0601	389	JZ CLDIS ; YES - CLEAR DISPLAY AND TERMINATE COMMAND
000D	FE11	390	CPI COMMA ; WAS LAST CHARACTER ',' ?
000F	C2FF01	391	JNZ ERR ; NO - DISPLAY ERROR MSG. AND TERMINATE COMMAND
0002	CD0F02	392	CALL IXTRG ; YES - ADVANCE REGISTER POINTER TO
		393	;/NEXT REGISTER
		394	;/ANY MORE REGISTERS ?
		395	TRUE EXM05 ; YES - CONTINUE PROCESSING WITH NEXT REGISTER
0005	DAAF00	396+	JC EXM05
0008	C30601	397	JMP CLDIS ; NO - CLEAR DISPLAY AND TERMINATE COMMAND
		398 ;	
		399 ;	*****
		400 ;	
		401 ;	FUNCTION: GOCMD - EXECUTE USER PROGRAM
		402 ;	INPUTS: NONE
		403 ;	OUTPUTS: NONE
		404 ;	CALLS: DISPC, RDKBD, CLEAR, GTHEX, EPR, OUTPT
		405 ;	DESTROYS: A, B, C, D, E, H, L, F/F'S
		406 ;	
		407	GOCMD:
0008	CD0E01	408	CALL DISPC ; DISPLAY USER PROGRAM COUNTER
000E	CD0002	409	CALL RDKBD ; READ FROM KEYBOARD
00E1	FE10	410	CPI PERIO ; IS CHARACTER A PERIOD ?
00E3	CAFA00	411	JZ G10 ; YES - GO EXECUTE THE COMMAND
		412	;/NO - ARG - CHARACTER IS STILL IN A
00E5	32FE20	413	STA IBUFF ; REPLACE CHARACTER IN INPUT BUFFER
00E9	CD0001	414	CALL CLEAR ; CLEAR DISPLAY
00EC	0300	415	MVI B, R0FLD ; ARG - USE ADDRESS FIELD
00EE	CD0002	416	CALL GTHEX ; GET HEX DIGITS
00F1	FE10	417	CPI PERIO ; WAS LAST CHARACTER A PERIOD ?
00F3	C2FF01	418	JNZ ERR ; NO - DISPLAY ERROR MSG. AND TERMINATE COMMAND
00F5	EB	419	XCHG ; PUT HEX VALUE FROM GTHEX TO H & L
00F7	22F220	420	SHLD PSAY ; HEX VALUE IS NEW USER PC
		421	G10:
00FA	CD0D01	422	CALL CLEAR ; CLEAR DISPLAY
00FD	21E703	423	LXI H, EXMSG ; GET ADDRESS OF EXECUTION MESSAGE IN H & L
0100	0E04	424	MVI C, 4 ; LOAD C WITH COUNT OF 4
0102	CD0E02	425	CALL OUTPT ; DISPLAY EXECUTION MESSAGE
0105	C32B03	426	JMP RSTOR ; RESTORE USER REGISTERS INCL. PROGRAM COUNTER
		427	;/I.E. BEGIN EXECUTION OF USER PROGRAM
		428 ;	
		429 ;	*****
		430 ;	
		431 ;	FUNCTION SSTEP - SINGLE STEP (EXECUTE ONE USER INSTRUCTION)
		432 ;	INPUTS: NONE
		433 ;	OUTPUTS: NONE
		434 ;	CALLS: DISPC, RDKBD, CLEAR, GTHEX, EPR
		435 ;	DESTROYS: A, B, C, D, E, H, L, F/F'S
		436 ;	
		437	SSTEP:
0100	CD0E01	438	CALL DISPC ; DISPLAY USER PROGRAM COUNTER
0108	CD0002	439	CALL RDKBD ; READ FROM KEYBOARD

ORIGINAL PAGE IS
OF POOR QUALITY

IS15-11 8080/8085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 2

LOC	OBJ	LINE	SOURCE STATEMENT
010E	FE10	440	CPI PERIO ; WAS CHARACTER A PERIOD ?
0110	CP0501	441	JZ CLDIS ; YES - CLEAR DISPLAY AND TERMINATE COMMAND
0113	FE11	442	CPI COMMA ; WAS LAST CHARACTER A COMMA ?
0115	CR2001	443	JZ STP20 ; YES - GO SET TIMER
		444	; NO - CHARACTER FROM KEYBOARD WAS NEITHER PERIOD OR COMMA
0118	32FE20	445	STA Ibuff ; REPLACE THE CHARACTER IN THE INPUT BUFFER
011B	CD0D01	446	CALL CLEAR ; CLEAR DISPLAY
011E	CD0D02	447	CALL GTHex ; GET HEX DIGITS - WERE ANY DIGITS RECEIVED ?
		448	FALSE ERR ; NO - DISPLAY ERROR MSG. AND TERMINATE COMMAND
0121	D2FF01	449+	JNC ERR
0124	EB	450	XCHG ; HEX VALUE FROM GTHex TO H & L
0125	22F220	451	SHLD PSW ; HEX VALUE IS NEW USER PC
0128	FE10	452	CPI PERIO ; WAS LAST CHARACTER FROM GTHex A PERIOD ?
012A	CR0601	453	JZ CLDIS ; YES - CLEAR DISPLAY AND TERMINATE COMMAND
		454	; NO - MUST HAVE BEEN A COMMA
		455	STP20:
012D	3AF120	456	LDA ISW ; GET USER INTERRUPT MASK
0130	E500	457	ANI 00H ; KEEP INTERRUPT STATUS
0132	32FD20	458	STA TEMP ; SAVE USER INTERRUPT STATUS IN TEMP
0135	2AF220	459	LHLD PSW ; GET USER PC
0138	7E	460	MOV R,M ; GET USER INSTRUCTION
0139	FEF3	461	CPI (DI) ; DI INSTRUCTION ?
013B	C24C01	462	JNZ STP21 ; NO
013E	AF	463	MRA R ; YES - RESET USER INTERRUPT STATUS
013F	C34901	464	JMP STP22
		465	STP21:
0142	F2F8	466	CPI (EI) ; EI INSTRUCTION ?
0144	C24C01	467	JNZ STP23 ; NO
0147	3E00	468	MVI R,00H ; YES - SET USER INTERRUPT STATUS
		469	STP22:
0149	32FD20	470	STA TEMP ; SAVE NEW USER INTERRUPT STATUS
		471	STP23:
014C	C32B03	472	JMP RSTOR ; RESTORE USER REGISTERS
		473 ;	
		474	STP25 ; BRANCH HERE WHEN TIMER INTERRUPTS AFTER
		475	; /ONE USER INSTRUCTION
014F	F5	476	PUSH PSW ; SAVE PSW
0150	3AFF20	477	LDA USCSR ; GET USER IMAGE OF WHAT'S IN CSR
0153	E53F	478	ANI 3FH ; CLEAR 2 HIGH ORDER BITS
0155	F1	479	POP PSW ; RETRIEVE PSW
0156	22EF20	480	SHLD LSW ; SAVE H & L
0159	E1	481	POP H ; GET USER PROGRAM COUNTER FROM TOP OF STACK
015A	22F220	482	SHLD PSW ; SAVE USER PC
015D	F5	483	PUSH PSW
015E	E1	484	POP H
015F	22ED20	485	SHLD FSW ; SAVE FLIP/FLOPS AND A REGISTER
0162	210000	486	LXI H,0 ; CLEAR H & L
0165	29	487	DAD SP ; GET USER STACK POINTER
0166	22F420	488	SHLD SSW ; SAVE USER STACK POINTER
0169	21ED20	489	LXI H,BSW+1 ; SET MONITOR STACK POINTER FOR
016C	F9	490	SPHL ; /SAVING REMAINING USER REGISTERS
016D	C5	491	PUSH B ; SAVE B & C
016E	D5	492	PUSH D ; SAVE D & E
016F	29	493	RIM ; GET USER INTERRUPT MASK
0170	E507	494	ANI 07H ; KEEP MASK BITS

ORIGINAL PAGE IS
OF POOR QUALITY

1315-11 0000/2005 MACRO ASSEMBLER, V2.0

MONITOR PAGE 10

LOC	OBJ	LINE	SOURCE STATEMENT
0172	24F020	495	LXI H,TEMP ; GET USER INTERRUPT STATUS
0175	86	496	ORA M ; OR IT INTO MASK
0176	32F120	497	STA ISAV ; SAVE INTERRUPT STATUS AND MASK
0179	3E8E	498	MVI A,UNMSK ; UNMASK INTERRUPTS FOR MONITOR USE
017B	30	499	SIM
017C	C30801	500	JMP SSTEP ; GO GET READY FOR ANOTHER INSTRUCTION
		501 ;	
		502 ;	*****
		503 ;	
		504 ;	FUNCTION: SUBST - SUBSTITUTE MEMORY
		505 ;	INPUTS: NONE
		506 ;	OUTPUTS: NONE
		507 ;	CALLS: CLEAR,GTHEX,UPRAD,UPDAT,EPR
		508 ;	DESTROYS: A,B,C,D,E,H,L,F/F'S
		509 ;	
		510	SUBST:
017F	C0CD01	511	CALL CLEAR ; CLEAR THE DISPLAY
0182	24EF03	512	LXI H,MSG ; DISPLAY SUBSTITUTE MESSAGE
0185	0E04	513	MVI C,4
0187	C0FE82	514	CALL OUTPT
018A	C00D02	515	CALL GTHEX ; GET HEX DIGITS - WERE ANY DIGITS RECEIVED ?
		516	FALSE EPR ; NO - DISPLAY ERROR MSG. AND TERMINATE COMMAND
018D	D2FF01	517+	JNC ERR
0190	EB	518	XCHG ; ASSIGN HEX VALUE RETURNED BY GTHEX TO
0191	22FE20	519	SHLD CURAD ; / CURRENT ADDRESS
		520	SUB05:
0194	FE11	521	CPI COMMA ; WAS ',' THE LAST CHARACTER FROM KEYBOARD ?
0196	C2C501	522	JNZ SUB15 ; NO - GO TERMINATE THE COMMAND
0199	0E00	523	MVI B,NODOT ; ARG - NO DOT IN ADDRESS FIELD
019B	C06F03	524	CALL UPRAU ; UPDATE ADDRESS FIELD OF DISPLAY
019E	2AF620	525	LHLD CURAD ; GET CURRENT ADDRESS IN H & L
01A1	7E	526	MOV A,M ; GET DATA BYTE POINTED TO BY CURRENT ADDRESS
01A2	32F820	527	STA CURDT ; STORE DATA BYTE AT CURRENT DATA
01A5	C07C03	528	CALL UPDAT ; UPDATE DATA FIELD OF DISPLAY
01A8	0601	529	MVI B,DYFLD ; ARG - USE DATA FIELD
01AA	C08D02	530	CALL GTHEX ; GET HEX DIGITS - WERE ANY HEX DIGITS RECEIVED ?
01AD	F5	531	PUSH PSW ; (SAVE LAST CHARACTER)
		532	FALSE SUB10 ; NO - LEAVE DATA UNCHANGED AT CURRENT ADDRESS
01AE	D2BA01	533+	JNC SUB10
01B1	2AF620	534	LHLD CURAD ; YES - GET CURRENT ADDRESS IN H & L
01B4	73	535	MOV M,E ; STORE NEW DATA AT CURRENT ADDRESS
		536	; MAKE SURE DATA WAS ACTUALLY STORED IN CASE
		537	;/CURRENT ADDRESS IS IN ROM OR IS NON-VOLITILE
01B5	7B	538	MOV A,E ; DATA TO A FOR COMPARISON
01B6	BE	539	CMP M ; WAS DATA STORED CORRECTLY
01B7	C2FF01	540	JNZ ERR ; NO - DISPLAY ERROR MSG. AND TERMINATE COMMAND
		541	SUB10:
01BA	2AF620	542	LHLD CURAD ; INCREMENT CURRENT ADDRESS
01BD	23	543	INX H
01BE	22F620	544	SHLD CURAD
01C1	F1	545	POP PSW ; RETRIEVE LAST CHARACTER
01C2	C39401	546	JMP SUB05 ;
		547	SUB15:
01C5	FE10	548	CPI PERIO ; WAS LAST CHARACTER A PERIOD ?
01C7	C2FF01	549	JNZ ERR ; NO - DISPLAY ERROR MSG. AND TERMINATE COMMAND

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 9080/9085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 11

LOC	OBJ	LINE	SOURCE STATEMENT
010A	020601	550	JMP CLDIS ; YES - CLEAR DISPLAY AND TERMINATE COMMAND
		551 ;	
		552 ;	
		553 ;	*****
		554 ;	
		555 ;	UTILITY ROUTINES
		556 ;	
		557 ;	*****
		558 ;	
		559 ;	FUNCTION: CLEAR - CLEAR DISPLAY
		560 ;	INPUTS: NONE
		561 ;	OUTPUTS: NONE
		562 ;	CALLS: OUTPT
		563 ;	DESTROYES: A,B,C,D,E,H,L,F/F'S
		564 ;	DESCRIPTION: CLEAR SENDS BLANK CHARACTERS TO BOTH THE ADDRESS FIELD
		565 ;	AND THE DATA FIELD OF THE DISPLAY. IF THE DOT FLAG IS
		566 ;	SET THEN A DOT WILL APPEAR AT THE RIGHT EDGE OF THE
		567 ;	ADDRESS FIELD
		568 ;	
		569	CLEAR:
010D	210803	570	LXI H,BLANKS ; ARG - ADDRESS OF BLANKS FOR DISPLAY
010D	0E08	571	MVI C,8 ; LOAD C WITH CHARACTER COUNT
01D2	0D9E02	572	CALL OUTPT ; OUTPUT BLANKS TO ADDRESS FIELD
01D5	C9	573	RET ; RETURN
		574 ;	
		575 ;	*****
		576 ;	
		577 ;	FUNCTION: CLDIS - CLEAR DISPLAY AND TERMINATE COMMAND
		578 ;	INPUTS: NONE
		579 ;	OUTPUTS: NONE
		580 ;	CALLS: CLEAR
		581 ;	DESTROYES: A,B,C,D,E,H,L,F/F'S
		582 ;	DESCRIPTION: CLDIS IS JUMPED TO BY COMMAND ROUTINES WISHING TO
		583 ;	TERMINATE NORMALLY. CLDIS CLEARS THE DISPLAY AND
		584 ;	BRANCHES TO THE COMMAND RECOGNIZER.
		585 ;	
		586	CLDIS:
01D6	0C0D01	587	CALL CLEAR ; CLEAR THE DISPLAY
01D9	C36490	588	JMP CMDND / GO GET ANOTHER COMMAND
		589 ;	
		590 ;	*****
		591 ;	
		592 ;	FUNCTION: CLDST - COLD START
		593 ;	INPUTS: NONE
		594 ;	OUTPUTS: NONE
		595 ;	CALLS: NOTHING
		596 ;	DESTROYES: A
		597 ;	DESCRIPTION: CLDST IS JUMPED TO BY THE MAIN COLD START PROCEDURE
		598 ;	COMPLETES THE COLD START INITIALIZATION.
		599 ;	AND JUMPS BACK TO THE MAIN COLD START PROCEDURE
		600 ;	
		601	CLDST:
01DC	3E90	602	MVI A,URTPC ; LOAD UART PORT CONTROL BYTE
01DE	0200	603	OUT CONCP ; OUTPUT PORT CONTROL BYTE
01E9	3E93	604	MVI A,URTSFC ; LOAD UART SYSTEM CONTROL BYTE

ORIGINAL PAGE IS
OF POOR QUALITY

IS15-II 6060/6065 MACRO ASSEMBLER V3.0

MONITOR PAGE 10

LSC OBJ	LINE	SOURCE STATEMENT
01E2 03DE	605	OUT CONSCP ; OUTPUT SYSTEM CONTROL BYTE TO SYSTEM PORT
	606	;
01E4 21E920	607	LXI H,MNSTK ; LOAD HL WITH STACK START ADDRESS
01E7 F9	608	SPHL ; STACK POINTER SET UP
	609	;
01E8 030500	610	JMP CLDBK ; BACK TO MAIN PROCEDURE
	611	;
	612	*****
	613	;
	614	FUNCTION: DISPC - DISPLAY PROGRAM COUNTER
	615	INPUTS: NONE
	616	OUTPUTS: NONE
	617	CALLS: UPDAD,UPDOT
	618	DESTROYES: A,B,C,D,E,H,L,F/F'S
	619	DESCRIPTION: DISPC DISPLAYS THE USER PROGRAM COUNTER IN THE ADDRESS
	620	FIELD OF THE DISPLAY,WITH A DOT AT THE RIGHT EDGE
	621	OF THE FIELD. THE BYTE OF DATA ADDRESSED BY THE PROGRAM
	622	COUNTER IS DISPLAYED IN THE DATA FIELD OF THE DISPLAY.
	623	;
	624	DISPC:
01EB 000001	625	CALL CLEAR ; CLEAR DISPLAY
01EE 2AF220	626	LHLD PSRV ; GET USER PROGRAM COUNTER
01F1 22F620	627	SHLD CURAD ; MAKE IT THE CURRENT ADDRESS
01F4 7E	628	MOV A,M ; GET THE INSTRUCTION AT THAT ADDRESS
01F5 32F820	629	STA CURDT ; MAKE IT THE CURRENT DATA
01F8 006F03	630	CALL UPDAD ; UPDATE ADDRESS FIELD OF DISPLAY
01FB 007003	631	CALL UPDOT ; UPDATE DATA FIELD OF DISPLAY
01FE 09	632	RET ; RETURN
	633	;
	634	*****
	635	;
	636	FUNCTION: EPR - DISPLAY ERROR MESSAGE
	637	INPUTS: NONE
	638	OUTPUTS: NONE
	639	CALLS: OUTPT
	640	DESTROYES: A,B,C,D,E,H,L,F/F'S
	641	DESCRIPTION: EPR IS JUMPED TO BY COMMAND ROUTINES WISHING TO
	642	TERMINATE BECAUSE OF AN ERROR.
	643	ERR OUTPUTS AN ERROR MESSAGE TO THE DISPLAY AND
	644	BRANCHES TO THE COMMAND RECOGNIZER.
	645	;
	646	ERR:
01FF 21E303	647	LXI H,ERMSG ; ARG - ADDRESS OF ERROR MESSAGE
0202 0E04	648	MVI C,4 ; LOAD 4 COUNT
0204 009E02	649	CALL OUTPT ; OUTPUT ERROR MESSAGE TO ADDRESS FIELD
0207 009103	650	CALL SECOT ; DELAY FOR 2 SECONDS
	651	;
020A 03E400	652	JMP CMND ; GO GET A NEW COMMAND
	653	;
	654	*****
	655	;
	656	FUNCTION: GTHX - GET HEX DIGITS
	657	INPUTS: NONE
	658	OUTPUTS: A - LAST CHARACTER READ FROM KEYBOARD
	659	DE - HEX DIGITS FROM KEYBOARD EVALUATED MODULO 16

ORIGINAL PAGE NO
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 13

LOC	OBJ	LINE	SOURCE STATEMENT
		660 ;	CARRY - SET IF AT LEAST ONE VALID HEX DIGIT WAS READ
		661 ;	- RESET OTHERWISE
		662 ;	CALLS: ROKBD, INSDG, HXDSP, OUTPT
		663 ;	DESTROYS: A, B, C, D, E, H, L, F/F'S
		664 ;	DESCRIPTION: GTHEX ACCEPTS A STRING OF HEX DIGITS FROM THE KEYBOARD,
		665 ;	DISPLAYS THEM AS THEY ARE RECEIVED, AND RETURNS THEIR
		666 ;	VALUE AS A 16 BIT INTEGER. IF MORE THAN 4 HEX DIGITS
		667 ;	ARE RECEIVED, ONLY THE LAST 4 ARE USED.
		668 ;	THE LAST FOUR HEXIDEcimal
		669 ;	DIGITS ARE DISPLAYED IN THE ADDRESS FIELD OF THE
		670 ;	DISPLAY. A CHARACTER WHICH IS NOT
		671 ;	A HEX DIGIT TERMINATES THE STRING AND IS RETURNED AS
		672 ;	AN OUTPUT OF THE FUNCTION. IF THE TERMINATOR IS NOT
		673 ;	A PERIOD OR A COMMA THEN ANY HEX DIGITS WHICH MAY HAVE
		674 ;	BEEN RECEIVED ARE CONSIDERED TO BE INVALID. THE
		675 ;	FUNCTION RETURNS A FLAG INDICATING WHETHER OR NOT ANY
		676 ;	VALID HEX DIGITS WERE RECEIVED.
		677 ;	
		678	GTHEX:
0200	0E00	679	MVI C,0 ; RESET HEX DIGIT FLAG
020F	110000	680	LXI D,0 ; SET HEX VALUE TO ZERO
0212	05	681	PUSH B ; SAVE BC
0213	05	682	PUSH D ; SAVE HEX VALUE
		683	GTH05:
0214	0C0002	684	CALL ROKBD ; READ KEYBOARD
0217	FE10	685	CPI 10H ; IS CHARACTER A HEX DIGIT ?
0219	023702	686	JNC GTH20 ; NO - GO CHECK FOR TERMINATOR
		687	; YES - ARG - NEW HEX DIGIT IS IN A
021C	01	688	POP D ; ARG - RETRIEVE HEX VALUE
021D	008602	689	CALL INSDG ; INSERT NEW DIGIT IN HEX VALUE
0220	C1	690	POP B ; RETRIEVE DISPLAY FLAG
0221	0E01	691	MVI C,1 ; SET HEX DIGIT FLAG
		692	; / (I.E. A HEX DIGIT HAS BEEN READ)
0223	05	693	PUSH B ; SAVE BC
0224	05	694	PUSH D ; SAVE HEX VALUE
0225	78	695	MOV A,B ; TEST DISPLAY FLAG
0226	0F	696	RRC ; SHOULD ADDRESS FIELD OF DISPLAY BE USED ?
0227	023102	697	JNC GTH10 ; YES - USE HEX VALUE AS IS
		698	; NO - ONLY LOW ORDER BYTE OF HEX VALUE SHOULD
		699	; / BE USED FOR DATA FIELD OF DISPLAY
022A	53	700	MOV D,E ; PUT LOW ORDER BYTE OF HEX VALUE IN D
022B	000003	701	CALL DSPDT2
022E	031402	702	JMP GTH05
		703	GTH10:
		704	; ARG - HEX VALUE TO BE EXPANDED IS IN D & E
0231	007203	705	CALL DSPDT4 ; GO DO 2 CHARACTER OUTPUT
		706	;
0234	031402	707	JMP GTH05 ; GO GET NEXT CHARACTER
		708	GTH20:
		709	; LAST CHARACTER WAS NOT A HEX DIGIT
0237	01	710	POP D ; RETRIEVE HEX VALUE
0238	FE11	711	CPI COMMA ; WAS LAST CHARACTER A COMMA ?
023A	0A4002	712	JZ GTH25 ; YES - READY TO RETURN
023D	FE10	713	CPI PERIO ; NO - WAS LAST CHARACTER A PERIOD ?
023F	0A4002	714	JZ GTH25 ; YES - READY TO RETURN

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0

MONITOR PAGE 14

LOC	OBJ	LINE	SOURCE STATEMENT
		715	; NO - INVALID TERMINATOR - IGNORE ANY HEX DIGITS READ
0242	110000	716	LXI D,0 ; SET HEX VALUE TO ZERO
0245	C30903	717	JMP RETF ; RETURN FALSE
		718	GTH25:
0248	C1	719	POP B ; POP BC
0249	47	720	MOV B,A ; SAVE LAST CHARACTER
024A	79	721	MOV A,C ; SHIFT HEX DIGIT FLAG TO
024B	0F	722	RRC ; /CARRY BIT
024C	78	723	MOV A,B ; RESTORE LAST CHARACTER
024D	C9	724	RET ; RETURN
		725	;
		726	;*****
		727	;
		728	; FUNCTION: HXDSP - EXPAND HEX DIGITS FOR DISPLAY
		729	; INPUTS: DE - 4 HEX DIGITS
		730	; OUTPUTS: HL - ADDRESS OF OUTPUT BUFFER
		731	; CALLS: NOTHING
		732	; DESTROYS: A,H,L,F/P'S
		733	; DESCRIPTION: HXDSP EXPANDS EACH INPUT BYTE TO 2 BYTES IN A FORM
		734	; SUITABLE FOR DISPLAY BY THE OUTPUT ROUTINES. EACH INPUT
		735	; BYTE IS DIVIDED INTO 2 HEX DIGITS. EACH HEX DIGIT IS
		736	; STORED IN THE OUTPUT BUFFER. THE FUNCTION RETURNS THE
		737	; ADDRESS OF THE OUTPUT BUFFER.
		738	;
		739	HXDSP:
024E	7A	740	MOV A,D ; GET FIRST DATA BYTE
024F	0F	741	RRC ; CONVERT 4 HIGH ORDER BITS
0250	0F	742	RRC ; /TO A SINGLE CHARACTER
0251	0F	743	RRC
0252	0F	744	RRC
0253	21F920	745	LXI H,OBUFF ; GET ADDRESS OF OUTPUT BUFFER
0256	CD7402	746	CALL HEXASC ; CONVERT BITS TO ASCII CODE IN A
0259	77	747	MOV M,A ; STORE CHARACTER IN OUTPUT BUFFER
025A	7A	748	MOV A,D ; GET FIRST DATA BYTE AND CONVERT 4 LOW ORDER
025B	23	749	INX H ; NEXT BUFFER POSITION
025C	CD7402	750	CALL HEXASC
025F	77	751	MOV M,A ; STORE CHARACTER IN BUFFER
0260	7B	752	HXDSP2: MOV A,E ; GET SECOND DATA BYTE
0261	0F	753	RRC ; CONVERT 4 HIGH ORDER BITS
0262	0F	754	RRC ; /TO A SINGLE CHARACTER
0263	0F	755	RRC
0264	0F	756	RRC
0265	23	757	INX H ; NEXT BUFFER POSITION
0266	CD7402	758	CALL HEXASC
0269	77	759	MOV M,A ; STORE CHARACTER IN BUFFER
026A	7B	760	MOV A,E ; GET SECOND DATA BYTE AND CONVERT LOW ORDER
026B	23	761	INX H ; NEXT BUFFER POSITION
026C	CD7402	762	CALL HEXASC
026F	77	763	MOV M,A ; STORE CHARACTER IN BUFFER
0270	21F920	764	LXI H,OBUFF ; RETURN ADDRESS OF OUTPUT BUFFER IN H & L
0273	C9	765	RET ; RETURN
		766	;
		767	;*****
		768	;
		769	; FUNCTION: HEXASC

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 2080/5085 MACRO ASSEMBLER V3.0

MONITR PAGE 15

LDC OBJ	LINE	SOURCE STATEMENT
	770 ;	INPUTS: HEX DIGIT IN ACCUMULATOR
	771 ;	OUTPUTS: ASCII CODE IN A
	772 ;	CALLS: NOTHING
	773 ;	DESTROYS: A,F/F'S
	774 ;	DESCRIPTION: HEXASC CONVERTS A HEX NIBBLE IN A TO AN ASCII CODE
	775 ;	
	776	HEXASC:
0274 E60F	777	ANI 0FH ; CLEAR UPPER BITS TO ZERO
0275 FE0A	778	CPI 0AH ; TEST BITS IF A-F
0278 DA7E02	779	JC HEX05 ; JUMP IF 0-9
027B C537	780	ADI 037H ; ADD 37 TO FORM ASCII
027D C9	781	RET ; RETURN
	782	HEX05:
027E F630	783	ORI 070H ; OR IN A 30 TO 0-9
0280 C9	784	RET ; RETURN
	785 ;	
	786 ;	*****
	787 ;	
	788 ;	FUNCTION: ININT ~ INPUT INTERRUPT PROCESSING
	789 ;	INPUTS: NONE
	790 ;	OUTPUTS: NONE
	791 ;	CALLS: NOTHING
	792 ;	DESTROYS: NOTHING
	793 ;	DESCRIPTION: ININT IS ENTERED BY MEANS OF AN INTERRUPT VECTOR (IV?)
	794 ;	
	795	ININT:
0281 E5	796	PUSH H ; SAVE H & L
0282 F3	797	PUSH PSW ; SAVE F/F'S & REGISTER A
	798 ;	***** INTERRUPT CODE GOES HERE
0283 F1	799	POP PSW ; RESTORE F/F'S & REGISTER A
0284 E1	800	POP H ; RESTORE H & L
0285 C9	801	RET ; RETURN
	802 ;	
	803 ;	*****
	804 ;	
	805 ;	FUNCTION: INSDG ~ INSERT HEX DIGIT
	806 ;	INPUTS: A - HEX DIGIT TO BE INSERTED
	807 ;	DE - HEX VALUE
	808 ;	OUTPUTS: DE - HEX VALUE WITH DIGIT INSERTED
	809 ;	CALLS: NOTHING
	810 ;	DESTROYES: A,F/F'S
	811 ;	DESCRIPTION: INSDG SHIFTS THE CONTENTS OF D & E LEFT 4 BITS
	812 ;	(1 HEX DIGIT) AND INSERTS THE HEX DIGIT IN THE LOW
	813 ;	ORDER DIGIT POSITION OF THE RESULT. A IS ASSUMED TO
	814 ;	CONTAIN A SINGLE HEX DIGIT IN THE LOW ORDER 4 BITS AND
	815 ;	ZEROS IN THE HIGH ORDER 4 BITS.
	816 ;	
	817	INSDG:
0286 EB	818	XCHG ; PUT D & E IN H & L
0287 29	819	DAD H ; SHIFT H & L LEFT 4 BITS
0288 29	820	DAD H
0289 29	821	DAD H
028A 29	822	DAD H
028B 95	823	ADD L ; INSERT LOW ORDER DIGIT
028C 6F	824	MOV L,A ; BY ADDING L+A = A

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080 8085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 16

LOC	OBJ	LINE	SOURCE STATEMENT
029D	EB	825	MOVG ; PUT H & L BACK IN D & E
029E	C9	826	RET ; RETURN
		827 ;	
		828 ;	*****
		829 ;	
		830 ;	FUNCTION: NXTRG - ADVANCE REGISTER POINTER TO NEXT REGISTER
		831 ;	INPUTS: NONE
		832 ;	OUTPUTS: CARRY - 1 IF POINTER IS ADVANCED SUCCESSFULLY
		833 ;	- 0 OTHERWISE
		834 ;	CALLS: NOTHING
		835 ;	DESTROYS: A, F/F'S
		836 ;	DESCRIPTION: IF THE REGISTER POINTER POINTS TO THE LAST REGISTER IN
		837 ;	THE EXAMINE REGISTER SEQUENCE, THE POINTER IS NOT
		838 ;	CHANGED AND THE FUNCTION RETURNS FALSE. IF THE REGISTER
		839 ;	POINTER DOES NOT POINT TO THE LAST REGISTER THEN THE
		840 ;	POINTER IS ADVANCED TO THE NEXT REGISTER IN THE SEQUENCE
		841 ;	AND THE FUNCTION RETURNS TRUE.
		842 ;	
		843	NXTRG:
029F	3AFD20	844	LDA RPTR ; GET REGISTER POINTER
0292	FE0C	845	CPI NUMRG-1 ; DOES POINTER POINT TO LAST REGISTER ?
0294	D20803	846	JNC RETF ; YES - UNABLE TO ADVANCE POINTER - RETURN FALSE
0297	3C	847	INR A ; NO - ADVANCE REGISTER POINTER
0298	32FD20	848	STA RPTR ; SAVE REGISTER POINTER
029B	C30E03	849	JMP RETT ; RETURN TRUE
		850 ;	
		851 ;	*****
		852 ;	
		853 ;	FUNCTION: OUTPT - OUTPUT CHARACTERS TO DISPLAY
		854 ;	INPUTS:
		855 ;	C - CHARACTER COUNT
		856 ;	HL - ADDRESS OF CHARACTERS TO BE OUTPUT
		857 ;	CALLS: NOTHING
		858 ;	DESTROYS: A, B, C, D, E, H, L, F/F'S
		859 ;	DESCRIPTION: OUTPT SENDS CHARACTERS TO THE DISPLAY. THE ADDRESS
		860 ;	OF THE CHARACTERS IS RECEIVED AS AN ARGUMENT
		861 ;	
		862	OUTPT:
029E	1507	863	MVI E, 7 ; CLEAR E TO 7 FOR ZERO CHARACTER POSITION
02A0	56	864	OUT05: MOV D, H ; LOAD CHARACTER IN (HL)
02A1	CDA802	865	CALL DISP1 ; CALL DISPLAY ROUTINE FOR 1 CHARACTER
02A4	1D	866	DCR E ; DECREMENT E FOR POSITION COUNT
02A5	23	867	INX H ; INCREMENT ADDRESS REGISTER HL TO NEXT CHARACTER
02A6	00	868	DCR C ; DECREMENT CHARACTER COUNT
02A7	C2A002	869	JNZ OUT05 ; LOOP BACK IF NOT DONE DISPLAY
02AA	C9	870	RET ; RETURN
		871 ;	*****
		872 ;	PRO-LOG 7303 DISPLAY OUTPUT CODE
02A8	7A	873	DISP1: MOV A, D ; OUTPUT DISPLAY DATA
02AC	F600	874	ORI 080H ; TURN ON BIT 8 FOR PRO-LOG
02AE	D3D0	875	OUT 0D0H ; PORT D0, SEND DATA
02B0	7B	876	MOV A, E ; OUTPUT DISPLAY ADDRESS, SET INCTIP HIGH
02B1	E6F7	877	ANI 0F7H ; AND TO FORM CODE 11110111
02B3	D3D1	878	OUT 0D1H ; PORT D1, SEND POSITION TO D1
02B5	7B	879	MOV A, E ; SET NOT NR LCN

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8860/9885 MACRO ASSEMBLER, V2.0

MONTR PAGE 17

LOC	OBJ	LINE	SOURCE STATEMENT
02B6	F600	880	ORI 06H ; OR IN A 00001000
02B8	07D1	881	OUT 0D1H ; PORT D1
02BA	78	882	MOV A,E
02BB	E6F7	883	ANI 0F7H ; AND IN A 11110111
02BD	03D1	884	OUT 0D1H ; PORT D1
02BF	C9	885	RET ; RETURN
		886 ;	
		887 ;	*****
		888 ;	
		889 ;	FUNCTION: RDKBD - READ KEYBOARD
		890 ;	INPUTS: NONE
		891 ;	OUTPUTS: A - CHARACTER READ FROM KEYBOARD
		892 ;	CALLS: NOTHING
		893 ;	DESTROYS: R,H,L,F/S
		894 ;	DESCRIPTION: RDKBD DETERMINES WHETHER OR NOT THERE IS A CHARACTER IN
		895 ;	THE INPUT BUFFER. IF NOT, THE FUNCTION ENABLES
		896 ;	INTERRUPTS AND LOOPS UNTIL THE INPUT INTERRUPT
		897 ;	ROUTINE STORES A CHARACTER IN THE BUFFER. WHEN
		898 ;	THE BUFFER CONTAINS A CHARACTER, THE FUNCTION FLAGS
		899 ;	THE BUFFER AS EMPTY AND RETURNS THE CHARACTER
		900 ;	AS OUTPUT.
		901 ;	
		902	RDKBD:
		903 ;	*****
		904 ;	PRO-LOG 7303 KEYBOARD INPUT DRIVER
02C0	D5	905	PUSH D ; SCAN KEYBOARD BY ENABLING
02C1	C5	906	PUSH B ; ONE COLUMN AT A TIME AND
		907 ;	
02C2	1688	908	MYI D,88H ; LOOKING FOR A KEY CLOSURE, 10001000
02C4	7A	909	NXTCOL: MOV A,D
02C5	07	910	RLC ; ROTATE A LEFT AND SET CARRY
02C6	57	911	MOV D,A ; SAVE SHIFTED PATTERN
02C7	E60F	912	ANI 0FH ; AND IN A 00001111
02C9	03D0	913	OUT 0D0H ; PORT D0
02CB	06D0	914	IN 0D0H ; PORT D0
02CD	E63F	915	ANI 3FH ; AND IN A 00111111
02CF	EAC402	916	JPE NXTCOL
02D2	5F	917	MOV E,A ; LOAD E WITH INPUT
02D3	0A0303	918	CALL MSEC20 ; WAIT 20 MSEC
02D6	06D0	919	IN 0D0H ; PORT D0
02D8	E63F	920	ANI 3FH ; AND IN A 00111111
02DA	A8	921	XRA E ; EOR E INTO ACC
02DB	C2C402	922	JNZ NXTCOL ; IF NOT SAME IGNORE AND NEXT COLUMN
02DE	010000	923	LXI B,0 ; CLEAR BC
02E1	7A	924	MOV A,D ; LOAD PATTERN
02E2	1F	925	COLCNT: RAR ; CONVERT COLUMN OUTPUT TO A COUNT K
02E3	0A0A02	926	JC ROWCNT ; JUMP ON CARRY TO ROW COUNT
02E6	A4	927	INR B
02E7	C3E202	928	JMP COLCNT
02EA	78	929	ROWCNT: MOV A,E ; CONVERT ROW INPUT TO A COUNT J
02EB	1F	930	RAR
02EC	0C	931	INR C
02ED	0AF302	932	JC ENCODE ; JUMP ON CARRY TO ENCODE
02F0	030902	933	JMP ROWCNT+1
02F3	00	934	ENCODE: DCR C ; DECREMENT REGISTER C

ORIGINAL PAGE IS
OF POOR QUALITY

1515-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITR PAGE 19

LOC	OBJ	LINE	SOURCE STATEMENT
02F4	78	935	MOV A,B ; OAD ACC WITH ROM
02F5	C9FE02	936	JZ RKEXIT
02F9	C604	937	ADDR0W: ADI 04H ; IF J.NE.0 ADD J*M (# OF COLS)
02FA	00	938	DCR C
02FB	C2F602	939	JNZ ADDR0W
		940	RKEXIT:
02FE	F5	941	PUSH PSH
02FF	E5	942	PUSH H
0300	CD9703	943	CALL MSEC2H
0303	E1	944	POP H
0304	F1	945	POP PSH
0305	C1	946	POP B ; RESTORE ALL REGISTERS
0306	D1	947	POP D ; POP STACK FOR DE
0307	C9	948	RET ; RETURN
		949 ;	
		950 ;	*****
		951 ;	
		952 ;	FUNCTION: RETF - RETURN FALSE
		953 ;	INPUTS: NONE
		954 ;	OUTPUTS: CARRY = 0 (FALSE)
		955 ;	CALLS: NOTHING
		956 ;	DESTROYS: CARRY
		957 ;	DESCRIPTION: RETF IS JUMPED TO BY FUNCTIONS WISHING TO RETURN FALSE
		958 ;	RETF RESETS CARRY TO 0 AND RETURNS TO THE CALLER OF
		959 ;	THE ROUTINE INVOKING RETF.
		960 ;	
		961	RETF:
0308	37	962	STC ; SET CARRY TRUE
0309	3F	963	CMC ; COMPLEMENT CARRY TO MAKE IT FALSE
030A	C9	964	RET ; RETURN
		965 ;	
		966 ;	*****
		967 ;	
		968 ;	FUNCTION RETT - RETURN TRUE
		969 ;	INPUTS: NONE
		970 ;	OUTPUTS: CARRY = 1 (TRUE)
		971 ;	CALLS: NOTHING
		972 ;	DESTROYS: CARRY
		973 ;	DESCRIPTION: RETT IS JUMPED TO BY ROUTINES WISHING TO RETURN TRUE
		974 ;	RETT SETS CARRY TO 1 AND RETURNS TO THE CALLER OF
		975 ;	THE ROUTINE INVOKING RETT.
		976 ;	
		977	RETT:
030B	37	978	STC ; SET CARRY TRUE
030C	C9	979	RET ; RETURN
		980 ;	
		981 ;	*****
		982 ;	
		983 ;	FUNCTION: RGL0C - GET REGISTER SAVE LOCATION
		984 ;	INPUTS: NONE
		985 ;	OUTPUTS: HL - REGISTER SAVE LOCATION
		986 ;	CALLS: NOTHING
		987 ;	DESTROYS: B,C,H,L,F'S
		988 ;	DESCRIPTION: RGL0C RETURNS THE SAVE LOCATION OF THE REGISTER
		989 ;	INDICATED BY THE CURRENT REGISTER POINTER VALUE

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
		990 ;	
		991 RGLOC:	
0300	2AFD20	992	LHLD RGPTR ; GET REGISTER POINTER.
0310	2600	993	MVI H,0 ; /IN H & L
0312	013004	994	LXI B,RGTBL ; GET REGISTER SAVE LOCATION TABLE ADDRESS
0315	09	995	DAD B ; POINTER INDEXES TABLE
0316	6E	996	MOV L,H ; GET LOW ORDER BYTE OF REGISTER SAVE LOC.
0317	2620	997	MVI H,(RMTST SHR 8) ; GET HIGH ORDER BYTE OF
		998	; /REGISTER SAVE LOCATION
0319	C9	999	RET ; RETURN
		1000 ;	
		1001 ;	*****
		1002 ;	
		1003 ;	FUNCTION: RGNAM - DISPLAY REGISTER NAME
		1004 ;	INPUTS: NONE
		1005 ;	OUTPUTS: NONE
		1006 ;	CALLS: OUTPT
		1007 ;	DESTROYS: A,B,C,D,E,H,L,F/F'S
		1008 ;	DESCRIPTION: RGNAM DISPLAYS, IN THE ADDRESS FIELD OF THE DISPLAY,
		1009 ;	THE REGISTER NAME CORRESPONDING TO THE CURRENT
		1010 ;	REGISTER POINTER VALUE
		1011 ;	
		1012 RGNAM:	
031A	2AFD20	1013	LHLD RGPTR ; GET REGISTER POINTER
031D	2600	1014	MVI H,0
031F	29	1015	DAD H ; MULTIPLY POINTER VALUE BY 4
0320	29	1016	DAD H ; /REGISTER NAME TABLE HAS 4 BYTE ENTRIES)
0321	010004	1017	LXI B,RMTBL ; GET ADDRESS OF START OF REGISTER NAME TABLE
0324	09	1018	DAD B ; RRG - ADD TABLE ADDRESS TO POINTER
		1019	; /ADDRESS OF APPROPRIATE REGISTER NAME IN H & L
0325	0E04	1020	MVI C,4
0327	CD9E02	1021	CALL OUTPT ; OUTPUT REGISTER NAME TO ADDRESS FIELD
032A	C9	1022	RET ; RETURN
		1023 ;	
		1024 ;	*****
		1025 ;	
		1026 ;	FUNCTION: RSTOR - RESTOR USER REGISTERS
		1027 ;	INPUTS: NONE
		1028 ;	OUTPUTS: NONE
		1029 ;	CALLS: NOTHING
		1030 ;	DESTROYS: A,B,C,D,E,H,L,F/F'S
		1031 ;	DESCRIPTION: RSTOR RESTORES ALL CPU REGISTERS, FLIP/FLOPS,
		1032 ;	INTERRUPT STATUS, INTERRUPT MASK, STACK POINTER
		1033 ;	AND PROGRAM COUNTER FROM THEIR RESPECTIVE SAVE
		1034 ;	LOCATIONS IN MEMORY. BY RESTORING THE PROGRAM
		1035 ;	COUNTER, THE ROUTINE EFFECTIVELY TRANSFERS CONTROL TO
		1036 ;	THE ADDRESS IN THE PROGRAM COUNTER SAVE LOCATION.
		1037 ;	
		1038 ;	THE TIMING OF THIS ROUTINE IS CRITICAL TO THE
		1039 ;	CORRECT OPERATION OF THE SINGLE STEP ROUTINE.
		1040 ;	IF ANY MODIFICATION CHANGES THE NUMBER OF CPU
		1041 ;	STATES NEEDED TO EXECUTE THIS ROUTINE THEN THE
		1042 ;	TIMER VALUE MUST BE ADJUSTED BY THE SAME NUMBER.
		1043 ;	
		1044 ;	THIS IS ALSO THE ENTRY POINT FOR THE TTY MONITOR

ORIGINAL PAGE IS
OF POOR QUALITY

IS15-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITR PAGE 20

LOC	OBJ	LINE	SOURCE STATEMENT
		1045 ;	TO RESTORE REGISTERS.
		1046 ;	
		1047	RESTOR.
0328	3AF120	1048	LDA ISAV ; GET USER INTERRUPT MASK
032E	F610	1049	ORI 1SH ; ENABLE SETTING INTERRUPT MASK AND
		1050	; /RESET IV3C FLIP FLOP
0330	30	1051	SIM ; RESTORE USER INTERRUPT MASK
		1052 ;	RESTORE USER INTERRUPT STATUS
0331	3AF120	1053	LDA ISAV ; GET USER INTERRUPT MASK
0334	E608	1054	ANI 08H ; SHOULD USER INTERRUPTS BE ENABLED ?
0336	CA3D03	1055	JZ RSR05 ; NO - LEAVE INTERRUPTS DISABLED
0339	FB	1056	EI ; YES - ENABLE INTERRUPTS FOR USER PROGRAM
033A	C34103	1057	JMP RSR10
		1058	RSR05:
033D	37	1059	STC ; DUMMY INSTRUCTIONS - WHEN SINGLE STEP ROUTINE
033E	D24103	1060	JNC RSR10 ; /IS BEING USED, THE TIMER IS RUNNING AND
		1061	; /EXECUTE TIME FOR THIS ROUTINE MUST
		1062	; /VARY.
		1063	RSR10:
0341	21E920	1064	LXI H,MNSTK ; SET MONITOR STACK POINTER TO START OF STACK
0344	F9	1065	SPHL ; /WHICH IS ALSO END OF REGISTER SAVE AREA
0345	D1	1066	POP D ; RESTORE REGISTERS
0346	C1	1067	POP B
0347	F1	1068	POP PSW
0349	3AF420	1069	LHLD SSRV ; RESTORE USER STACK POINTER
034B	F9	1070	SPHL
034C	3AF220	1071	LHLD PSRV
034F	E5	1072	PUSH H ; PUT USER PROGRAM COUNTER ON STACK
0350	3AEF20	1073	LHLD LSRV ; RESTORE H & L REGISTERS
0353	C9	1074	RET ; RETURN
		1075 ;	
		1076 ;	*****
		1077 ;	
		1078 ;	FUNCTION: SETRG - SET REGISTER POINTER
		1079 ;	INPUTS: NONE
		1080 ;	OUTPUTS: CARRY - SET IF CHARACTER FROM KEYBOARD IS A REGISTER DESIGNATOR
		1081 ;	RESET OTHERWISE
		1082 ;	CALLS: ROKBD
		1083 ;	DESTROYS: A,B,C,H,L,P/F'S
		1084 ;	DESCRIPTION: SETRG READS A CHARACTER FROM THE KEYBOARD. IF THE
		1085 ;	CHARACTER IS A REGISTER DESIGNATOR, IT IS CONVERTED TO
		1086 ;	THE CORRESPONDING REGISTER POINTER VALUE. THE POINTER IS
		1087 ;	SAVED, AND THE FUNCTION RETURNS 'TRUE' OTHERWISE, THE
		1088 ;	FUNCTION RETURNS 'FALSE'
		1089 ;	
		1090	SETRG:
0354	CDC002	1091	CALL ROKBD ; READ FROM KEYBOARD
0357	FE10	1092	CPI 10H ; IS CHARACTER A DIGIT ?
0359	020003	1093	JNC RETF ; NO - RETURN FALSE - CHARACTER IS NOT A
		1094	; /REGISTER DESIGNATOR
035C	D603	1095	SUI 3 ; YES - TRY TO CONVERT REGISTER DESIGNATOR TO
		1096	; / INDEX INTO REGISTER POINTER TABLE
		1097	; WAS CONVERSION SUCCESSFUL ?
035E	D40003	1099	JC RETF ; NO - RETURN FALSE
0361	4F	1099	MOV C,A ; INDEX TO B & C

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
0362	0E00	1100	MVI B,0 ;
0364	21F02	1101	LXI H,ROPTB ; GET ADDRESS OF REGISTER POINTER TABLE
0367	09	1102	DAD B ; INDEX POINTS INTO TABLE
0368	7E	1103	MOV A,M ; GET REGISTER POINTER FROM TABLE
0369	32FD20	1104	STA ROPTR ; SAVE REGISTER POINTER
036C	C30803	1105	JMP RETT ; RETURN TRUE
		1106 ;	
		1107 ;	*****
		1108 ;	
		1109 ;	FUNCTION: UPDAD - UPDATE ADDRESS FIELD OF DISPLAY
		1110 ;	INPUTS: NONE
		1111 ;	OUTPUTS: NONE
		1112 ;	CALLS: HXDSP,OUTPT
		1113 ;	DESTROYS: A,B,C,D,E,H,L,F/'S
		1114 ;	DESCRIPTION: UPDAD UPDATES THE ADDRESS FIELD OF THE DISPLAY USING
		1115 ;	THE CURRENT ADDRESS.
		1116 ;	
		1117	UPDAD:
036F	2AF620	1118	LHLD CURAD ; GET THE CURRENT ADDRESS
0372	EB	1119	MCHG ; ARG - PUT CURRENT ADDRESS IN D & E
		1120	DSPDT4 ; ENTRY POINT FOR 4 CHARACTER ADDRESS OUTPUT
0373	CD4E02	1121	CALL HXDSP ; EXPAND CURRENT ADDRESS FOR DISPLAY
		1122	; ARG - ADDRESS OF EXPANDED ADDRESS IS IN H & L
0376	0E04	1123	MVI C,04H ; LOAD 4 CHARACTER COUNT
0379	CD9E02	1124	CALL OUTPT ; OUTPUT CURRENT ADDRESS TO ADDRESS FIELD
037B	C3	1125	RET ; RETURN
		1126 ;	
		1127 ;	*****
		1128 ;	
		1129 ;	FUNCTION: UPDDT - UPDATE DATA FIELD OF DISPLAY
		1130 ;	INPUTS: NONE
		1131 ;	OUTPUTS: NONE
		1132 ;	CALLS: HXDSP,OUTPT
		1133 ;	DESTROYS: A,B,C,D,E,H,L,F/'S
		1134 ;	DESCRIPTION: UPDDT UPDATES THE DATA FIELD OF THE DISPLAY USING
		1135 ;	THE CURRENT DATA BYTE.
		1136 ;	
		1137	UPDDT:
037C	3AF820	1138	LDA CURDT ; GET CURRENT DATA
037F	57	1139	MOV D,A ; ARG - PUT CURRENT DATA IN D
		1140	DSPDT2 ; ENTRY POINT FOR 2 CHARACTER DISPLAY
0380	CD4E02	1141	CALL HXDSP ; EXPAND CURRENT DATA FOR DISPLAY
		1142	; ARG - ADDRESS OF EXPANDED DATA IS IN H & L
0383	0E02	1143	MVI C,2
0385	1E02	1144	MVI E,2 ; SET TO 6TH SLOT
0387	CD9002	1145	CALL OUT05 ; ENTER OUTPUT ROUTINE
038A	C3	1146	RET ; RETURN
		1147 ;	
		1148 ;	*****
		1149 ;	
		1150 ;	FUNCTION: DELAYS
		1151 ;	INPUTS: NONE
		1152 ;	OUTPUTS: NONE
		1153 ;	CALLS: NOTHING
		1154 ;	DESTROYS: A,B,C,F/'S

ORIGINAL PAGE IS
OF POOR QUALITY

IS15-11 0000/0005 MACRO ASSEMBLER, V3.0 MONITR PAGE 22

LOC	OBJ	LINE	SOURCE STATEMENT
		1155 ;	DESCRIPTION. DELAYS IS THE DELAY TIME FUNCTION BLOCK
		1156 ;	FROM THE PRO-LOG MONITOR. THE ENTRY POINTS
		1157 ;	INCLUDE 1 SEC, 200 MSEC, 100 MSEC, 20 MSEC, AND 1 MSEC.
		1158 ;	THE DELAY ROUTINES ARE CALLED AND RETURN
		1159 ;	WHEN DELAY FINISHES.
		1160 ;	
036B	010100	1161	DELAY: LXI B,1 ; LOAD BC WITH A 1
036E	C3A603	1162	JMP MSECOT ; GO TO 1 MSEC DELAY LOOP
		1163	SECOT:
0391	01E803	1164	LXI B,03EEH ; LOAD BC WITH 1000 DECIMAL
0394	C3A603	1165	JMP MSECOT
0397	01C800	1166	MSEC2H: LXI B,0200H ; LOAD BC WITH 200 DECIMAL
039A	C3A603	1167	JMP MSECOT
039D	016400	1168	MSEC1H: LXI B,0064H ; LOAD BC WITH 100 DECIMAL
03A0	C3A603	1169	JMP MSECOT
03A3	011400	1170	MSEC20: LXI B,0014H ; LOAD BC WITH 20 DECIMAL
03A6	00	1171	MSECOT: DCX B ; DELAY N MILLISECDS
03A7	C5	1172	PUSH B ; WHERE N16=(BC)
03A8	0ED8	1173	MVI C,00BH
03AA	0D	1174	DELENT: DCR C
03AB	C2AA03	1175	JNZ DELENT
03AE	C1	1176	POP B ; POP BC OFF STACK
03AF	78	1177	MOV A,B
03B0	81	1178	ORA C
03B1	C2A603	1179	JNZ MSECOT ; JUMP BACK TO MSECOT IF NOT DONE
03B4	C9	1180	RET ; RETURN
		1181 ;	
		1182 ;	*****
		1183 ;	
		1184 ;	MONITOR TABLES
		1185 ;	
		1186 ;	*****
		1187 ;	
		1188 ;	COMMAND TABLE
		1189 ;	COMMAND CHARACTERS AS RECEIVED FROM KEYBOARD
		1190	CMDTB:
03B5	12	1191	DB 12H ; GO COMMAND
03B6	13	1192	DB 13H ; SUBSTITUTE COMMAND
03B7	17	1193	DB 17H ; EXAMINE REGISTERS COMMAND
03B8	15	1194	DB 15H ; SINGLE STEP COMMAND
		1195 ;	DB 10H ; EXEC KEY
		1196 ;	DB 11H ; NEXT KEY
		1197 ;	DB 16H ; DUMMY
03B9	14	1198	DB 14H ; TTY MONITOR
		1199 ;	
0005		1200	MUNC EQU *-CMDTB ; NUMBER OF COMMANDS
		1201 ;	
		1202 ;	*****
		1203 ;	
		1204 ;	COMMAND ROUTINE ADDRESS TABLE
		1205 ;	(MUST BE IN REVERSE ORDER OF COMMAND TABLE)
		1206	CMDAD:
		1207 ;	DW DUMMY COMMAND 1
		1208 ;	DW DUMMY COMMAND 2
		1209 ;	DW DUMMY COMMAND 3

ORIGINAL PAGE IS
OF POOR QUALITY

1515-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITR PAGE 23

LOC	OBJ	LINE	SOURCE STATEMENT
01EA	4904	1210	DN GO ; TTY MONITOR
03BC	0001	1211	DN SSTEP ; ADDRESS OF SINGLE STEP ROUTINE
03BE	9E00	1212	DN EXAM ; ADDRESS OF EXAMINE REGISTERS ROUTINE
03C0	7F01	1213	DN SUBST ; ADDRESS OF SUBSTITUTE MEMORY ROUTINE
03C2	0000	1214	DN GOCMD ; ADDRESS OF GO ROUTINE
		1215 ;	
		1216 ;	*****
		1217 ;	
		1218	DSPNUM: ; DISPLAY TABLE FOR HEX DIGITS
		1219 ;	
		1220	DSPTB: ; TABLE FOR TRANSLATING CHARACTERS FOR OUTPUT
		1221 ;	
		1222 ;	DISPLAY
		1223 ;	FORMAT CHARACTERS
		1224 ;	=====
		1225 ;	
		1226 ;	PRO-LOG 7303 USES 8 BIT ASCII, 7 BIT PLUS BIT 8 = 1
		1227 ;	
00B0		1228	ZERO EQU 000H
03C4	B0	1229	DB 000H ; 0
03C5	B1	1230	DB 001H ; 1
03C6	B2	1231	DB 002H ; 2
03C7	B3	1232	DB 003H ; 3
03C8	B4	1233	DB 004H ; 4
00B5		1234	FIVE EQU 005H
03C9	B5	1235	DB 005H ; 5
03CA	B6	1236	DB 006H ; 6
03CB	B7	1237	DB 007H ; 7
00B8		1238	EIGHT EQU 008H
03CC	B8	1239	DB 008H ; 8
03CD	B9	1240	DB 009H ; 9
00C1		1241	LETRA EQU 001H
03CE	C1	1242	DB 001H ; A
00C2		1243	LETRB EQU 002H
03CF	C2	1244	DB 002H ; B
00C3		1245	LETRC EQU 003H
03D0	C3	1246	DB 003H ; C
00C4		1247	LETRD EQU 004H
03D1	C4	1248	DB 004H ; D
00C5		1249	LETRE EQU 005H
03D2	C5	1250	DB 005H ; E
00C6		1251	LETRF EQU 006H
03D3	C6	1252	DB 006H ; F
00C8		1253	LETRH EQU 008H
03D4	C8	1254	DB 008H ; H
00CC		1255	LETRL EQU 00CH
03D5	CC	1256	DB 00CH ; L
00D0		1257	LETRP EQU 000H
03D6	D0	1258	DB 000H ; P
00C9		1259	LETRI EQU 009H
03D7	C9	1260	DB 009H ; I
00D2		1261	LETRR EQU 002H
03D8	D2	1262	DB 002H ; R
00D3		1263	LETRS EQU 003H
03D9	D3	1264	DB 003H ; S

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8086/8085 MACRO ASSEMBLER, V3.0

MONTR PAGE 24

LOC	OBJ	LINE	SOURCE STATEMENT
0398		1265	BLANK EQU 0A0H
03DA A0		1266	DB 0A0H ; BLANK
		1267	;
		1268	*****
		1269	;
		1270	MESSAGES FOR OUTPUT TO DISPLAY
		1271	;
03DB A0		1272	BLANKS: DB BLANK, BLANK, BLANK, BLANK
03DC A0			
03DD A0			
03DE A0			
03DF A0		1273	BLANKS: DB BLANK, BLANK, BLANK, BLANK ; FOR ADDRESS OR DATA FIELD
03E0 A0			
03E1 A0			
03E2 A0			
03E3 2A		1274	ERMSG: DB '*', LETRE, LETRR, LETRR ; ERROR MESSAGE FOR ADDR. FIELD
03E4 C5			
03E5 D2			
03E6 D2			
03E7 C5		1275	EXMSG: DB LETRE, 'X', LETRE, 'Q' ; EXECUTION MESSAGE
03E8 5B			
03E9 C5			
03EA 51			
03EB 2A45584D		1276	XMSG: DB '*EXM' ; EXAMINE MESSAGE
03EF 2A53424D		1277	SMSG: DB '*SBM' ; SUBSTITUTE MEMORY
		1278	;
		1279	RDYTABLE FOR MONITOR PROMPT '@MONTR'
		1280	;
03F3 2A524541		1281	RDYTAB: DB '*READY *' ; MONITOR PROMPT
03F7 4459202A			
		1282	;
		1283	*****
		1284	;
		1285	RGPTB: ; REGISTER POINTER TABLE
		1286	;
		1287	THE ENTRIES IN THIS TABLE ARE IN THE SAME ORDER
		1288	AS THE REGISTER DESIGNATION KEYS ON THE KEYBOARD.
		1289	EACH ENTRY CONTAINS THE REGISTER POINTER VALUE WHICH
		1290	CORRESPONDS TO THE REGISTER DESIGNATOR. REGISTER
		1291	POINTER VALUES ARE USED TO POINT INTO THE REGISTER
		1292	NAME TABLE (NMTEL) AND REGISTER SAVE LOCATION
		1293	TABLE (RGTEL).
		1294	DB 6 ; INTERRUPT MASK
03FB 06			
03FC 09		1295	DB 9 ; SPH
03FD 0A		1296	DB 10 ; SPL
03FE 0B		1297	DB 11 ; PCH
03FF 0C		1298	DB 12 ; PCL
0400 07		1299	DB 7 ; H
0401 08		1300	DB 8 ; L
0402 00		1301	DB 0 ; A
0403 01		1302	DB 1 ; B
0404 02		1303	DB 2 ; C
0405 03		1304	DB 3 ; D
0406 04		1305	DB 4 ; E
0407 05		1306	DB 5 ; FLAGS

ORIGINAL PAGE IS
OF POOR QUALITY

1575-11 9090/9085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 25

LOC	OBJ	LINE	SOURCE STATEMENT
		1307 ;	
		1308 ;	
		1309 ;	
		1310	INTEL: ; REGISTER NAME TABLE
		1311	; NAMES OF REGISTERS IN DISPLAY FORMAT
0408	A0	1312	DB BLANK, BLANK, LETRA, BLANK ; A REGISTER
0409	A0		
040A	C1		
040B	A0		
040C	A0	1313	DB BLANK, BLANK, LETRB, BLANK ; B REGISTER
040D	A0		
040E	C2		
040F	A0		
0410	A0	1314	DB BLANK, BLANK, LETRC, BLANK ; C REGISTER
0411	A0		
0412	C3		
0413	A0		
0414	A0	1315	DB BLANK, BLANK, LETRD, BLANK ; D REGISTER
0415	A0		
0416	C4		
0417	A0		
0418	A0	1316	DB BLANK, BLANK, LETRE, BLANK ; E REGISTER
0419	A0		
041A	C5		
041B	A0		
041C	A0	1317	DB BLANK, BLANK, LETRF, BLANK ; FLAGS
041D	A0		
041E	C6		
041F	A0		
0420	A0	1318	DB BLANK, BLANK, LETRI, BLANK ; INTERRUPT MASK
0421	A0		
0422	C9		
0423	A0		
0424	A0	1319	DB BLANK, BLANK, LETRH, BLANK ; H REGISTER
0425	A0		
0426	C9		
0427	A0		
0428	A0	1320	DB BLANK, BLANK, LETRL, BLANK ; L REGISTER
0429	A0		
042A	CC		
042B	A0		
042C	D3	1321	DB LETRS, LETRP, LETRH, BLANK ; STACK POINTER HIGH ORDER BYTE
042D	D0		
042E	C3		
042F	A0		
0430	D3	1322	DB LETRS, LETRP, LETRL, BLANK ; STACK POINTER LOW ORDER BYTE
0431	D0		
0432	CC		
0433	A0		
0434	D0	1323	DB LETRP, LETRC, LETRH, BLANK ; PROGRAM COUNTER HIGH BYTE
0435	C2		
0436	C0		
0437	A0		
0438	D0	1324	DB LETRP, LETRC, LETRL, BLANK ; PROGRAM COUNTER LOW BYTE
0439	C3		

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 8880/8885 MACRO ASSEMBLER, V3.0

MONITR PAGE 27

LOC	OBJ	LINE	SOURCE STATEMENT
		1378 ;	
		1379 ;	MACROS USED IN THE TTY MONITOR ARE DEFINED IN THE KEYBOARD MONITOR.
		1380 ;	
		1381 ;	LIST OF FUNCTIONS
		1382 ;	=====
		1383 ;	
		1384 ;	GETCH
		1385 ;	-----
		1386 ;	
		1387 ;	DCMD
		1388 ;	GCMD
		1389 ;	ICMD
		1390 ;	MCMD
		1391 ;	SCMD
		1392 ;	XCMD
		1393 ;	-----
		1394 ;	
		1395 ;	CT
		1396 ;	CHVEN
		1397 ;	CO
		1398 ;	CROUT
		1399 ;	DELAY
		1400 ;	ECHO
		1401 ;	ERROR
		1402 ;	FRET
		1403 ;	GETCH
		1404 ;	GETHX
		1405 ;	GETNM
		1406 ;	HIL0
		1407 ;	NMOUT
		1408 ;	PRVAL
		1409 ;	REGDS
		1410 ;	RGADR
		1411 ;	SRET
		1412 ;	STHF0
		1413 ;	STHLF
		1414 ;	VALDG
		1415 ;	VALDL
		1416 ;	-----
		1417 ;	
		1418 ;	
		1419 ;	*****
		1420 ;	
		1421 ;	
		1422 ;	MONITOR EQUATES
		1423 ;	
		1424 ;	
		1425 ;	*****
		1426 ;	
		1427 ;	
		1428 ;	THIS TABLE IS CODED IN 7 BIT ASCII FOR TTY OR CRT
		1429 ;	
0013		1430 BRCHR EQU 10H ;	CODE FOR BREAK CHARACTER (ESCAPE)
0920		1431 BRTAB EQU 0920H ;	LOCATION OF START OF BRANCH TABLE IN ROM
0000		1432 CR EQU 00H ;	CODE FOR CARRIAGE RETURN

```

LOC  CBJ  LINE      SOURCE STATEMENT
001B      1433 ESC EQU 1BH ; CODE FOR ESCAPE CHARACTER
006F      1434 HCHAR EQU 0FH ; MASK TO SELECT LOWER HEX CHAR FROM BYTE
00FF      1435 INVRT EQU 0FFH ; MASK TO INVERT HALF BYTE FLAG
000A      1436 LF EQU 0AH ; CODE FOR LINE FEED
0000      1437 LOWER EQU 0 ; DENOTES LOWER HALF OF BYTE IN ICMD
          1438 ;LSGNON EQU --- ; LENGTH OF SIGNON MESSAGE - DEFINED LATER
          1439 ;MNSTK EQU --- ; START OF MONITOR STACK - DEFINED IN
          1440 ; ; /KEYBOARD MONITOR
          1441 ;NCMD5 EQU --- ; NUMBER OF VALID COMMANDS - DEFINED LATER
000F      1442 NEWLN EQU 0FH ; MASK FOR CHECKING MEMORY ADDR DISPLAY
007F      1443 PRY0 EQU 07FH ; MASK TO CLEAR PARITY BIT FROM CONSOLE CHAR
          1444 ;RANST EQU --- ; START ADDRESS OF RAM - DEFINED IN
          1445 ; ; KEYBOARD MONITOR
          1446 ;RTA25 EQU --- ; SIZE OF ENTRY IN RTAB TABLE
001B      1447 TERM EQU 1BH ; CODE FOR ICMD TERMINATING CHARACTER (ESCAPE)
00FF      1448 UPPER EQU 0FFH ; DENOTES UPPER HALF OF BYTE IN ICMD
          1449 ;
          1450 ; PRO-LOG FIXURES
000C      1451 CNIN EQU 00CH ; UART DATA INPUT PORT
000C      1452 CNOUT EQU 00CH ; UART DATA OUTPUT PORT
000D      1453 CSTAT EQU 000H ; CONSOLE STATUS PORT
000C      1454 CDATA EQU 00CH ; CONSOLE DATA PORT
000D      1455 CONCP EQU 000H ; CONSOLE CONTROL PORT
000E      1456 CONSCP EQU 00EH ; CONSOLE SYSTEM CONTROL PORT
0000      1457 URTPC EQU 000H ; UART PORT CONTROL WORD
0003      1458 URTPC EQU 00H ; UART SYSTEM PORT CONTROL WORD
0000      1459 TBE EQU 00H ; TERMINAL BUFFER EMPTY FLAG
0040      1460 RBF EQU 10H ; RECIEVE BUFFER FULL FLAG
          1461 ;
          1462 ;
          1463 ;*****
          1464 ;
          1465 ;
          1466 ; TTY MONITOR ENTRY POINT
          1467 ;
          1468 ;
          1469 ;*****
          1470 ;
          1471 ;
          1472 ;
          1473 ; PRINT SIGNON MESSAGE
          1474 ;
          1475 GO:
0449 219307 1476 LXI H,SGNON ; GET ADDRESS OF SIGNON MESSAGE
044C 0614    1477 MVI B,LSGNON ; COUNTER FOR CHARACTERS IN MESSAGE
          1478 MSG1:
044E 4E      1479 MOV C,M ; FETCH NEXT CHAR TO C REGISTER
044F CDF405 1480 CALL CO ; SEND IT TO THE CONSOLE
0452 23      1481 INX H ; POINT TO NEXT CHARACTER
0453 05      1482 DCR B ; DECREMENT BYTE COUNTER
0454 C24E04 1483 JNZ MSG1 ; RETURN FOR NEXT CHARACTER
          1484 ;
          1485 ;
          1486 ;*****
          1487 ;

```

```

LOC 081      LINE      SOURCE STATEMENT

1488 ;
1489 ;                COMMAND RECOGNIZING ROUTINE
1490 ;
1491 ;
1492 ;*****
1493 ;
1494 ; FUNCTION: GETCM
1495 ; INPUTS: NONE
1496 ; OUTPUTS: NONE
1497 ; CALLS: GETCH,ECHO,ERROR
1498 ; DESTROYS: A,B,C,H,L,F/F'S
1499 ; DESCRIPTION: GETCM RECIEVES AN INPUT CHARACTER FROM THE USER
1500 ;                AND ATTEMPTS TO LOCATE THIS CHARACTER IN ITS COMMAND
1501 ;                CHARACTER TABLE. IF SUCCESSFUL, THE ROUTINE
1502 ;                CORRESPONDING TO THIS CHARACTER IS SELECTED FROM
1503 ;                A TABLE OF COMMAND ROUTINE ADDRESSES, AND CONTROL
1504 ;                IS TRANSFERED TO THIS ROUTINE. IF THE CHARACTER
1505 ;                DOES NOT MATCH ANY ENTRIES, CONTROL IS PASSED TO
1506 ;                THE ERROR HANDLER.
1507 ;
1508 GETCM:
0457 21E920 1509 LXI H,MNSTK ; ALWAYS WANT TO RESET STACK PTR TO MONITOR
045A F9      1510 SPHL ; /STARTING VALUE SO ROUTINES NEEDN'T CLEAN UP
045B 0E2E     1511 MVI C,'.' ; PROMPT CHARACTER TO C
045D C00506 1512 CALL ECHO ; SEND PROMPT CHARACTER TO USER TERMINAL
045E C36304 1513 JMP GTC03 ; WANT TO LEAVE ROOM FOR RST BRANCH
1514 GTC03:
0463 C02006 1515 CALL GETCH ; GET COMMAND CHARACTER TO A
0466 C00506 1516 CALL ECHO ; ECHO CHARACTER TO USER
0469 79       1517 MOV A,C ; PUT COMMAND CHARACTER INTO ACCUMULATOR
046A 010600 1518 LXI B,NCMD5 ; C CONTAINS LOOP AND INDEX COUNT
046D 21B507 1519 LXI H,CTAB ; HL POINTS INTO COMMAND TABLE
1520 GTC05:
0470 BE       1521 CMP M ; COMPARE TABLE ENTRY AND CHARACTER
0471 0A7C04 1522 JZ GTC10 ; BRANCH IF EQUAL - COMMAND RECOGNIZED
0474 23       1523 INX H ; ELSE, INCREMENT TABLE POINTER
0475 0D       1524 DCR C ; DECREMENT LOOP COUNT
0476 C27004 1525 JNZ GTC05 ; BRANCH IF NOT AT TABLE END
0479 C31E06 1526 JMP ERROR ; ELSE, COMMAND CHARACTER IS ILLEGLE
1527 GTC10:
047C 21A707 1528 LXI H,CADR ; IF GOOD COMMAND, LOAD ADDRESS OF TABLE
1529 ; /OF COMMAND ROUTINE ADDRESSES
047F 05       1530 DAD B ; ADD WHAT IS LEFT OF LOOP COUNT
0480 09       1531 DAD B ; ADD AGAIN - EACH ENTRY IN CADR IS 2 BYTES LONG
0481 7E       1532 MOV A,M ; GET LSP OF ADDRESS OF TABLE ENTRY TO A
0482 23       1533 INX H ; POINT TO NEXT BYTE IN TABLE
0483 66       1534 MOV H,M ; GET MSP OF ADDRESS OF TABLE ENTRY TO H
0484 6F       1535 MOV L,A ; PUT LSP OF ADDRESS OF TABLE ENTRY INTO L
0485 E9       1536 PCHL ; NEXT INSTRUCTION COMES FROM COMMAND ROUTINE
1537 ;
1538 ;
1539 ;*****
1540 ;
1541 ;
1542 ;                COMMAND IMPLEMENTING ROUTINES

```


ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 20

LOC	OBJ	LINE	SOURCE STATEMENT
		1543 ;	
		1544 ;	
		1545 ;	*****
		1546 ;	
		1547 ;	
		1548 ;	FUNCTION: DCMD
		1549 ;	INPUTS: NONE
		1550 ;	OUTPUTS: NONE
		1551 ;	CALLS: ECHO, NMOUT, HILO, GETCM, CROUT, GETNM
		1552 ;	DESTROYS: A, B, C, D, E, H, L, F/F'S
		1553 ;	DESCRIPTION: DCMD IMPLEMENTS THE DISPLAY MEMORY (D) COMMAND
		1554 ;	
		1555	DCMD:
0486	0E02	1556	MVI C, 2 ; GET 2 NUMBERS FROM INPUT STREAM
0488	CD6806	1557	CALL GETNM
048B	D1	1558	POP D ; ENDING ADDRESS TO DE
048C	E1	1559	POP H ; STARTING ADDRESS TO HL
		1560	DCM05:
048D	CDFF05	1561	CALL CROUT ; ECHO CARRIAGE RETURN/LINE FEED
0490	7C	1562	MOV A, H ; DISPLAY ADDRESS OF FIRST LOCATION IN LINE
0491	CD0406	1563	CALL NMOUT
0494	7D	1564	MOV A, L ; ADDRESS IS 2 BYTES LONG
0495	CD0406	1565	CALL NMOUT
		1566	DCM10:
0498	0E20	1567	MVI C, ' '
049A	CD0506	1568	CALL ECHO ; USE BLANK AS SEPARATOR
049D	7E	1569	MOV A, H ; GET CONTENTS OF NEXT MEMORY LOCATION
049E	CD0406	1570	CALL NMOUT ; DISPLAY CONTENTS
04A1	CDAD06	1571	CALL HILO ; SEE IF ADDRESS OF DISPLAYED LOCATION IS
		1572	; /GREATER THAN OR EQUAL TO ENDING ADDRESS
		1573	FALSE DCM15 ; IF NOT, MORE TO DISPLAY
04A4	D2AD04	1574+	JNC DCM15
04A7	CDFF05	1575	CALL CROUT ; CARRIAGE RETURN/LINE FEED TO END LINE
04AA	C35704	1576	JMP GETCM ; ALL DONE
		1577	DCM15:
04AD	23	1578	INX H ; IF MORE TO GO, POINT TO NEXT LOC TO DISPLAY
04AE	7D	1579	MOV A, L ; GET LOW ORDER BITS OF NEW ADDRESS
04AF	EE0F	1580	ANI NEWLN ; SEE IF LAST HEX DIGIT OF ADDRESS DENOTES
		1581	; /START OF NEW LINE
04B1	C29904	1582	JNZ DCM10 ; NO - NOT AT END OF LINE
04B4	C38D04	1583	JMP DCM05 ; YES - START NEW LINE WITH ADDRESS
		1584 ;	
		1585 ;	
		1586 ;	*****
		1587 ;	
		1588 ;	
		1589 ;	FUNCTION: GCMO
		1590 ;	INPUTS: NONE
		1591 ;	OUTPUTS: NONE
		1592 ;	CALLS: ERROR, GETHX, RSTTF
		1593 ;	DESTROYS: A, B, C, D, E, H, L, F/F'S
		1594 ;	DESCRIPTION: GCMO IMPLEMENTS THE BEGIN EXECUTION (G) COMMAND.
		1595 ;	
		1596	GCMO:
04B7	CD3306	1597	CALL GETHX ; GET ADDRESS (IF PRESENT) FROM INPUT STREAM

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 8090/9085 MACRO ASSEMBLER, V3.0

MONITR PAGE 21

LOC	OBJ	LINE	SOURCE STATEMENT
		1598	FALSE GCM05 ; BRANCH IF NO NUMBER PRESENT
048A	02CC04	1599+	JNC GCM05
048D	7A	1600	MOV A,D ; ELSE, GET TERMINATOR
048E	FE00	1601	CPI CR ; SEE IF CARRIAGE RETURN
04C0	C21E06	1602	JNZ ERROR ; ERROR IF NOT PROPERLY TERMINATED
04C3	21F220	1603	LXI H,PSAV ; WANT NUMBER TO REPLACE SAVE PGM COUNTER
04C6	71	1604	MOV M,C
04C7	23	1605	INX H
04C9	70	1606	MOV M,B
04C9	C3D204	1607	JMP GCM10
		1608	GCM05:
04CC	7A	1609	MOV A,D ; IF NO STARTING ADDRESS, MAKE SURE THAT
04CD	FE00	1610	CPI CR ; /CARRIAGE RETURN TERMINATED COMMAND
04CF	C21E06	1611	JNZ ERROR ; ERROR IF NOT
		1612	GCM10:
04D2	C32903	1613	JMP RSTOR ; RESTORE REGISTERS AND BEGIN EXECUTION
		1614	; (RSTOR IS IN KEYBOARD MONITOR)
		1615 ;	
		1616 ;	
		1617 ;	*****
		1618 ;	
		1619 ;	
		1620 ;	FUNCTION: ICM0
		1621 ;	INPUTS: NONE
		1622 ;	OUTPUTS: NONE
		1623 ;	CALLS: ERROR, ECHO, GETCH, VALDL, VALDG, CNVBN, STHL, GETNM, CROUT
		1624 ;	DESTROYS: A,B,C,D,E,H,L,F/F'S
		1625 ;	DESCRIPTION: ICM0 IMPLEMENTS THE INSERT CODE INTO MEMORY (I) COMMAND.
		1626 ;	
		1627	ICM0:
04D5	0E01	1628	MVI C,1
04D7	CD6806	1629	CALL GETIN ; GET SINGLE NUMBER FROM INPUT STREAM
04DA	3EFF	1630	MVI A,UPPER
04DC	32FD20	1631	STA TEMP ; TEMP WILL HOLD THE UPPER/LOWER HALF BYTE FLAG
04DF	D1	1632	POP D ; ADDRESS OF START TO DE
		1633	ICM05:
04E0	CD2C06	1634	CALL GETCH ; GET A CHARACTER FROM INPUT STREAM
04E3	4F	1635	MOV C,A
04E4	C00506	1636	CALL ECHO ; ECHO IT
04E7	79	1637	MOV A,C ; PUT CHARACTER BACK INTO A
04E9	FE18	1638	CPI TERM ; SEE IF CHARACTER IS A TERMINATING CHARACTER
04EA	CA1605	1639	JZ ICM25 ; IF SO, ALL DONE ENTERING CHARACTERS
04ED	C07E07	1640	CALL VALDL ; ELSE, SEE IF VALID DELIMITER
		1641	TRUE ICM05 ; IF SO SIMPLY IGNORE THIS CHARACTER
04F0	DAE004	1642+	JC ICM05
04F3	CD6707	1643	CALL VALDG ; ELSE, CHECK TO SEE IF VALID HEX DIGIT
		1644	FALSE ICM20 ; IF NOT, BRANCH TO HANDLE ERROR CONDITION
04F5	D21005	1645+	JNC ICM20
04F9	C0EB05	1646	CALL CNVBN ; CONVERT DIGIT TO BINARY
04FC	4F	1647	MOV C,A ; MOVE RESULT TO C
04FD	C04407	1648	CALL STHL ; STORE IN APPROPRIATE HALF WORD
0500	3AFD20	1649	LDA TEMP ; GET HALF BYTE FLAG
0503	87	1650	CRA A ; SET F/F'S
0504	C20905	1651	JNZ ICM10 ; BRANCH IF FLAG SET FOR UPPER
0507	13	1652	INX D ; IF LOWER, INC ADDRESS OF BYTE TO STORE IN

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 32

LOC	OBJ	LINE	SOURCE STATEMENT
		1653	ICM10:
0508	EEFF	1654	XRI INVRT ; TOGGLE STATE OF FLAG
050A	32FD20	1655	STA TEMP ; PUT NEW VALUE OF FLAG BACK
050D	C3E004	1656	JMP ICM05 ; PROCESS NEXT DIGIT
		1657	ICM20:
0510	CD3907	1658	CALL STHF0 ; ILLEGAL CHARACTER
0513	C31E06	1659	JMP ERROR ; MAKE SURE ENTIRE BYTE FILLED THEN ERROR
		1660	ICM25:
0516	CD3907	1661	CALL STHF0 ; HERE FOR ESCAPE CHARACTER - INPUT IS DONE
0519	CDFF05	1662	CALL CROUT ; ADD CARRIAGE RETURN
051C	C35704	1663	JMP GETCM
		1664 ;	
		1665 ;	
		1666 ;	*****
		1667 ;	
		1668 ;	
		1669 ;	FUNCTION: MCM0
		1670 ;	INPUTS: NONE
		1671 ;	OUTPUTS: NONE
		1672 ;	CALLS: GETCM, HILO, GETNM
		1673 ;	DESTROYS: A, B, C, D, E, H, L, F/F'S
		1674 ;	DESCRIPTION: MCM0 IMPLEMENTS THE MODE DATA IN MEMORY (M) COMMAND.
		1675 ;	
		1676	MCM0:
051F	0E03	1677	MVI C, 3
0521	CD6806	1678	CALL GETNM ; GET 3 NUMBERS FROM INPUT STREAM
0524	C1	1679	POP B ; DESTINATION ADDRESS TO BC
0525	E1	1680	POP H ; ENDING ADDRESS TO HL
0526	D1	1681	POP D ; STARTING ADDRESS TO DE
		1682	MCM05:
0527	E5	1683	PUSH H ; SAVE ENDING ADDRESS
0528	62	1684	MOV H, D
0529	6B	1685	MOV L, E ; SOURCE ADDRESS TO HL
052A	7E	1686	MOV A, M ; GET SOURCE BYTE
052B	60	1687	MOV H, B
052C	69	1688	MOV L, C ; DESTINATION ADDRESS TO HL
052D	77	1689	MOV M, A ; MOVE BYTE TO DESTINATION
052E	03	1690	INX B ; INCREMENT DESTINATION ADDRESS
052F	78	1691	MOV A, B
0530	01	1692	ORA C ; TEST FOR DESTINATION ADDRESS OVERFLOW
0531	CA5704	1693	JZ GETCM ; IF SO, CAN TERMINATE COMMAND
0534	13	1694	INX D ; INCREMENT SOURCE ADDRESS
0535	E1	1695	POP H ; ELSE, GET BACK ENDING ADDRESS
0536	CDAD06	1696	CALL HILO ; SEE IF ENDING ADDR=SOURCE ADDR
		1697	FALSE GETCM ; IF NOT, COMMAND IS DONE
0539	D25704	1698+	JNC GETCM
053C	C32705	1699	JMP MCM05 ; MOVE ANOTHER BYTE
		1700 ;	
		1701 ;	
		1702 ;	*****
		1703 ;	
		1704 ;	
		1705 ;	FUNCTION: SCMD
		1706 ;	INPUTS: NONE
		1707 ;	OUTPUTS: NONE

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 8080/8085 MACRO ASSEMBLER, V1.0

MONITOR PAGE 22

LOC	OBJ	LINE	SOURCE STATEMENT
		1708	; CALLS: GETHX,GETCM,NNOUT,ECHO
		1709	; DESTROYS: A,B,C,D,E,H,L,F/F'S
		1710	; DESCRIPTION: SCMD IMPLEMENTS THE SUBSTITUTE INTO MEMORY (S) COMMAND.
		1711	;
		1712	SCMD:
053F	CD3306	1713	CALL GETHX ; GET A NUMBER, IF PRESENT, FROM INPUT
0542	C5	1714	PUSH B
0543	E1	1715	POP H ; GET NUMBER TO HL - DENOTES MEMORY LOCATION
		1716	SCM05:
0544	7A	1717	MOV A,D ; GET TERMINATOR
0545	FE20	1718	CPI ' ' ; SEE IF SPACE
0547	CA4F05	1719	JZ SCM10 ; YES - CONTINUE PROCESSING
0548	FE2C	1720	CPI ',' ; ELSE, SEE IF COMMA
054C	C25704	1721	JNZ GETCM ; NO - TERMINATE COMMAND
		1722	SCM10:
054F	7E	1723	MOV A,M ; GET CONTENTS OF SPECIFIED LOCATION TO A
0550	CD0406	1724	CALL NNOUT ; DISPLAY CONTENTS ON CONSOLE
0553	0E2D	1725	MVI C,'-'
0555	CD0506	1726	CALL ECHO ; USE DASH FOR SEPARATOR
0558	CD3306	1727	CALL GETHX ; GET NEW VALUE FOR MEMORY LOCATION, IF ANY
		1728	FALSE SCM15 ; IF NO VALUE PRESENT, BRANCH
055B	025F05	1729+	JNC SCM15
055E	71	1730	MOV M,C ; ELSE, STORE LOWER 8 BITS OF NUMBER ENTERED
		1731	SCM15:
055F	22	1732	INX H ; INCREMENT ADDRESS OF MEMORY LOCATION TO VIEW
0560	CD4405	1733	JMP SCM05
		1734	;
		1735	;
		1736	*****
		1737	;
		1738	;
		1739	; FUNCTION: XCMD
		1740	; INPUTS: NONE
		1741	; OUTPUTS: NONE
		1742	; CALLS: GETCH,ECHO,REGDS,GETCM,ERROR,PGADR,NNOUT,CROUT,GETHX
		1743	; DESTROYS: A,B,C,D,E,H,L,F/F'S
		1744	; DESCRIPTION: XCMD IMPLEMENTS THE REGISTER EXAMINE AND CHANGE (X)
		1745	COMMAND.
		1746	;
		1747	XCMD:
0563	CD2C06	1748	CALL GETCH ; GET REGISTER IDENTIFIER
0566	4F	1749	MOV C,A
0567	CD0506	1750	CALL ECHO ; ECHO IT
056A	73	1751	MOV A,C
056B	FE0D	1752	CPI CR
056D	C27605	1753	JNZ XCM05 ; BRANCH IF NOT CARRIAGE RETURN
0570	CDEF06	1754	CALL REGDS ; ELSE, DISPLAY REGISTER CONTENTS
0573	C35704	1755	JMP GETCM ; THEN TERMINATE COMMAND
		1756	XCM05:
0576	4F	1757	MOV C,A ; GET REGISTER IDENTIFIER TO C
0577	CD2007	1758	CALL PGADR ; CONVERT IDENTIFIER INTO RTAB TABLE ADDR
057A	C5	1759	PUSH B
057B	E1	1760	POP H ; PUT POINTER TO REGISTER ENTRY INTO HL
057C	0E2D	1761	MVI C,' '
057E	CD0506	1762	CALL ECHO ; ECHO SPACE TO USER

ORIGINAL PAGE IS
OF POOR QUALITY

1515-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 24

LOC	OBJ	LINE	SOURCE STATEMENT
0591	79	1763	MOV A,C
0592	32FD20	1764	STA TEMP ; PUT SPACE INTO TEMP AS DELIMITER
		1765	XCM10:
0595	3AFD20	1766	LDA TEMP ; GET TERMINATOR
0598	FE20	1767	CPI ' ' ; SEE IF BLANK
059A	CA9205	1768	JZ XCM15 ; YES - GO CHECK POINTER INTO TABLE
059D	FE2C	1769	CPI ',' ; NO - SEE IF COMMA
059F	C25704	1770	JNZ GETCH ; NO - MUST BE CARRIAGE RETURN TO END COMMAND
		1771	XCM15:
0592	7E	1772	MOV A,M
0593	87	1773	ORA A ; SET F/F'S
0594	C29005	1774	JNZ XCM18 ; BRANCH IF NOT AT END OF TABLE
0597	C0FF05	1775	CALL CROUT ; ELSE, OUTPUT CARRIAGE RETURN LINE FEED
059A	C35704	1776	JMP GETCH ; AND EXIT
		1777	XCM18:
059D	E5	1778	PUSH H ; PUT POINTER ON STACK
059E	5E	1779	MOV E,M
059F	1620	1780	MVI D,AMST SHR 8 ; ADDRESS OF SAVE LOCATION FROM TABLE
05A1	23	1781	INX H
05A2	46	1782	MOV B,M ; FETCH LENGTH FLAG FROM TABLE
05A3	D5	1783	PUSH D ; SAVE ADDRESS OF SAVE LOCATION
05A4	D5	1784	PUSH D
05A5	E1	1785	POP H ; MOVE ADDRESS TO HL
05A6	C5	1786	PUSH B ; SAVE LENGTH FLAG
05A7	7E	1787	MOV A,M ; GET 8 BITS OF REGISTER FROM SAVE LOCATION
05A8	C0D406	1788	CALL NMOUT ; DISPLAY IT
05AB	F1	1789	POP PSW ; GET BACK LENGTH FLAG
05AC	F5	1790	PUSH PSW ; SAVE IT AGAIN
05AD	87	1791	ORA A ; SET F/F'S
05AE	CA8605	1792	JZ XCM20 ; IF 8 BIT REGISTER, NOTHING MORE TO DISPLAY
05B1	2B	1793	DCX H ; ELSE, FOR 16 BIT REGISTER, GET LOWER 8 BITS
05B2	7E	1794	MOV A,M
05B3	C0D406	1795	CALL NMOUT ; DISPLAY THEM
		1796	XCM20:
05B6	0E2D	1797	MVI C,'-'
05B9	CD9506	1798	CALL ECHO ; USE DASH AS SEPARATOR
05BB	CD3306	1799	CALL GETHX ; SEE IF THERE IS A VALUE TO PUT INTO REGISTER
		1800	FALSE XCM30 ; NO - GO CHECK FOR NEXT REGISTER
05BE	D2D605	1801+	JNC XCM30
05C1	7A	1802	MOV A,D
05C2	32FD20	1803	STA TEMP ; ELSE, SAVE THE TERMINATOR FOR NOW
05C5	F1	1804	POP PSW ; GET BACK LENGTH FLAG
05C6	E1	1805	POP H ; PUT ADDRESS OF SAVE LOCATION INTO HL
05C7	87	1806	ORA A ; SET F/F'S
05C8	CACD05	1807	JZ XCM25 ; IF 8 BIT REGISTER, BRANCH
05CB	70	1808	MOV M,B ; SAVE UPPER 8 BITS
05CC	2B	1809	DCX H ; POINT TO SAVE LOCATION FOR LOWER 8 BITS
		1810	XCM25:
05CD	71	1811	MOV M,C ; STORE ALL OF 8 BIT OR LOWER 1/2 OF 16 BIT REG
		1812	XCM27:
05CE	110200	1813	LXI D,RTAB ; SIZE OF ENTRY IN RTAB TABLE
05D1	E1	1814	POP H ; POINTER INTO REGISTER TABLE RTAB
05D2	19	1815	DAD D ; ADD ENTRY SIZE TO POINTER
05D3	C28505	1816	JMP XCM10 ; DO NEXT REGISTER
		1817	XCM30:

ORIGINAL PAGE IS
OF POOR QUALITY

LOC	OBJ	LINE	SOURCE STATEMENT
05D6	7A	1818	MOV A,D ; GET TERMINATOR
05D7	32FD20	1819	STA TEMP ; SAVE IN MEMORY
05DA	01	1820	POP D ; CLEAR STACK OF LENGTH FLAG AND ADDRESS
05DB	01	1821	POP D ; /OF SAVE LOCATION
05DC	C3CE95	1822	JMP XCM27 ; GO INCREMENT REGISTER TABLE POINTER
		1823 ;	
		1824 ;	
		1825 ;	*****
		1826 ;	
		1827 ;	
		1828 ;	UTILITY ROUTINES
		1829 ;	
		1830 ;	
		1831 ;	*****
		1832 ;	
		1833 ;	
		1834 ;	FUNCTION: CI
		1835 ;	INPUTS: NONE
		1836 ;	OUTPUTS: A CHARACTER FROM TTY
		1837 ;	CALLS: NOTHING
		1838 ;	DESTROYS: A,F,F'S
		1839 ;	DESCRIPTION: CI WAITS UNTIL A CHARACTER HAS BEEN ENTERED AT THE
		1840 ;	TTY AND THEN RETURNS THE CHARACTER VIA THE H
		1841 ;	REGISTER, TO THE CALLING ROUTINE. THIS ROUTINE
		1842 ;	IS CALLED BY THE USER VIA A JUMP TABLE IN RAM.
		1843 ;	
		1844	CI:
05DF	DB0D	1845	IN CSTAT ; GET STATUS OF CONSOLE
05E1	E540	1846	ANI RBF ; CHECK FOR RECIEVER BUFFER FULL
05E3	CADF95	1847	JZ CI ; NOT READY YET - WAIT
05E6	D5DC	1848	IN CNIN ; READY, GET CHARACTER
05E8	E67F	1849	ANI 7FH ; AND IN 01111111 TO KILL PARITY BIT
05EA	C9	1850	RET ; THAT'S IT
		1851 ;	
		1852 ;	
		1853 ;	*****
		1854 ;	
		1855 ;	
		1856 ;	FUNCTION: CNVN
		1857 ;	INPUTS: C - ASCII CHARACTER '0' - '9' OR 'A' - 'F'
		1858 ;	OUTPUTS: A - 0 TO F HEX
		1859 ;	CALLS: NOTHING
		1860 ;	DESTROYS: A,F,F'S
		1861 ;	DESCRIPTION: CNVN CONVERTS THE ASCII REPRESENTATION OF A HEX
		1862 ;	DIGIT INTO ITS CORRESPONDING BINARY VALUE. CNVN
		1863 ;	DOES NOT CHECK THE VALIDITY OF ITS INPUT.
		1864 ;	
		1865	CNVN:
05EB	79	1866	MOV A,C
05EC	0630	1867	SUI '8' ; SUBTRACT CODE FOR '0' FROM ARGUMENT
05EE	FE08	1868	CPI 10 ; WANT TO TEST FOR RESULT 0 TO 9
05F0	F9	1869	RM ; IF SO, THEN ALL DONE
05F1	0607	1870	SUI 7 ; ELSE, RESULT BETWEEN 17 AND 23 DECIMAL
05F3	C9	1871	RET ; SO RETURN AFTER SUBTRACTING BIAS OF 7
		1872 ;	

ORIGINAL PAGE IS
OF POOR QUALITY

LSIS-II 8080/8085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 26

LDC OBJ	LINE	SOURCE STATEMENT
	1873 ;	
	1874 ;	*****
	1875 ;	
	1876 ;	
	1877 ;	FUNCTION: CO
	1878 ;	INPUTS: C - CHARACTER TO OUTPUT TO TTY
	1879 ;	OUTPUTS: C - CHARACTER OUTPUT TO TTY
	1880 ;	CALLS: NOTHING
	1881 ;	DESTROYS: A,F/F'S
	1882 ;	DESCRIPTION: CO SENDS ITS INPUT ARGUMENT TO THE TTY.
	1883 ;	
	1884 CO:	
05F4 0E00	1885	IN CSTAT ; GET STATUS OF CONSOLE
05F6 E300	1886	ANI TBE ; SEE IF TRANSMITTER READY
05F8 CAF405	1887	JZ CO ; NOT READY - WAIT
05FB 79	1888	MOV A,C ; LOAD A WITH CHARACTER
05FC D30C	1889	OUT COUT ; SEND CHARACTER
05FE C9	1890	RET ; ALL DONE
	1891 ;	
	1892 ;	
	1893 ;	*****
	1894 ;	
	1895 ;	
	1896 ;	FUNCTION CROUT
	1897 ;	INPUTS: NONE
	1898 ;	OUTPUTS: NONE
	1899 ;	CALLS: ECHO
	1900 ;	DESTROYS: A,B,C,F/F'S
	1901 ;	DESCRIPTION: CROUT SENDS A CARRIAGE RETURN (AND HENCE A LINE
	1902 ;	FEED) TO THE CONSOLE
	1903 ;	
	1904 CROUT:	
05FF 0E00	1905	MVI C,CR
0601 CD0506	1906	CALL ECHO
0604 C9	1907	RET
	1908 ;	
	1909 ;	
	1910 ;	*****
	1911 ;	
	1912 ;	
	1913 ;	FUNCTION: ECHO
	1914 ;	INPUTS: C - CHARACTER TO ECHO TO TERMINAL
	1915 ;	OUTPUTS: C - CHARACTER ECHOED TO TERMINAL
	1916 ;	CALLS: CO
	1917 ;	DESTROYS: A,B,F/F'S
	1918 ;	DESCRIPTION: ECHO TAKES A SINGLE CHARACTER AS INPUT AND, VIA
	1919 ;	THE MONITOR, SENDS THAT CHARACTER TO THE USER
	1920 ;	TERMINAL. A CARRIAGE RETURN IS ECHOED AS A CARRIAGE
	1921 ;	RETURN LINE FEED, AND AN ESCAPE CHARACTER IS ECHOED AS \$.
	1922 ;	
	1923 ECHO:	
0605 41	1924	MOV B,C ; SAVE ARGUMENT
0606 3E19	1925	MVI A,ESC
0608 B8	1926	CMP B ; SEE IF ECHOING AN ESCAPE CHARACTER
0609 C20E26	1927	JNZ ECH05 ; NO - BRANCH

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0

MONTR PAGE 27

LOC	OBJ	LINE	SOURCE STATEMENT
060C	0E24	1928	MVI C,'\$' ; YES - ECHO AS \$
		1929	ECH05:
060E	0DF405	1930	CALL C0 ; DO OUTPUT THROUGH MONITOR
0611	3E00	1931	MVI A,CR
0613	B8	1932	CMP B ; SEE IF CHARACTER ECHOED WAS A CARRIAGE RETURN
0614	021006	1933	JNZ ECH10 ; NO - NO NEED TO TAKE SPECIAL ACTION
0617	0E0A	1934	MVI C,LF ; YES - WANT TO ECHO LINE FEED, TOO
0619	0DF405	1935	CALL C0
		1936	ECH10:
061C	48	1937	MOV C,B ; RESTORE ARGUMENT
061D	C9	1938	RET ; RETURN
		1939 ;	
		1940 ;	
		1941 ;	*****
		1942 ;	
		1943 ;	
		1944 ;	FUNCTION: ERROR
		1945 ;	INPUTS: NONE
		1946 ;	OUTPUTS: NONE
		1947 ;	CALLS: ECHO CROUT,GETCH
		1948 ;	DESTROYS: A,B,C,F/F'S
		1949 ;	DESCRIPTION: ERROR PRINTS THE ERROR CHARACTER (CURRENTLY AN ASTERISK)
		1950 ;	ON THE CONSOLE, FOLLOWED BY A CARRIAGE RETURN LINE FEED,
		1951 ;	AND THEN RETURNS CONTROL TO THE COMMAND RECOGNIZER.
		1952 ;	
		1953	ERROR:
061E	0E2A	1954	MVI C,'*' ; LOAD ASTERISK INTO REGISTER C
0620	0D0506	1955	CALL ECHO ; SEND * TO CONSOLE
0623	0DFF05	1956	EXIT: CALL CROUT ; SKIP TO BEGINNING OF NEXT LINE
0626	C35704	1957	JMP GETCH ; TRY AGAIN FOR ANOTHER COMMAND
		1958 ;	
		1959 ;	
		1960 ;	*****
		1961 ;	
		1962 ;	
		1963 ;	FUNCTION: FRET
		1964 ;	INPUTS: NONE
		1965 ;	OUTPUTS: CARRY - ALWAYS 0
		1966 ;	CALLS: NOTHING
		1967 ;	DESTROYS: CARRY
		1968 ;	DESCRIPTION: FRET IS JUMPED TO BY ANY ROUTINE THAT WISHES TO
		1969 ;	INDICATE FAILURE ON RETURN. FRET SETS THE CARRY
		1970 ;	FALSE, DENOTING FAILURE, AND THEN RETURNS TO THE
		1971 ;	CALLER OF THE ROUTINE INVOKING FRET.
		1972 ;	
		1973	FRET:
0629	27	1974	STC ; FIRST SET CARRY TRUE
062A	3F	1975	CMC ; THEN COMPLEMENT IT TO MAKE IT FALSE
062B	C9	1976	RET ; RETURN APPROPRIATELY
		1977 ;	
		1978 ;	
		1979 ;	*****
		1980 ;	
		1981 ;	
		1982 ;	FUNCTION: GETCH

ORIGINAL PAGE IS
OF POOR QUALITY

1915-11 9080/9085 MACRO ASSEMBLER, V3.0

MONITOR PAGE 38

L3C C8J	LINE	SOURCE STATEMENT
	1983 ;	INPUTS: NONE
	1984 ;	OUTPUTS: D - NEXT CHARACTER IN INPUT STREAM
	1985 ;	CALLS: CI
	1986 ;	DESTROYS: D, E, F, F'S
	1987 ;	DESCRIPTION: GETCH RETURNS THE NEXT CHARACTER IN THE INPUT STREAM
	1988 ;	TO THE CALLING PROGRAM
	1989 ;	
	1990 ;	GETCH:
062C 000F05	1991	CALL CI ; GET CHARACTER FROM THE TERMINAL
062F E57F	1992	ANI PRY0 ; TURN OFF PARITY BIT IN CASE SET BY CONSOLE
0621 4F	1993	MOV C, A ; PUT VALUE IN C REGISTER FOR RETURN
0622 C9	1994	RET ; RETURN
	1995 ;	
	1996 ;	
	1997 ;	*****
	1998 ;	
	1999 ;	
	2000 ;	FUNCTION: GETHX
	2001 ;	INPUTS: NONE
	2002 ;	OUTPUTS: BC - 16 BIT INTEGER
	2003 ;	D - CHARACTER WHICH TERMINATED THE INTEGER
	2004 ;	CARRY - 1 IF FIRST CHARACTER NOT DELIMITER
	2005 ;	- 0 IF FIRST CHARACTER IS DELIMITER
	2006 ;	CALLS: ECHO, GETCH, VALDL, VALDG, CHVEN, ERROR
	2007 ;	DESTROYS: A, B, C, D, E, F, F'S
	2008 ;	DESCRIPTION: GETHX ACCEPTS A STRING OF HEX DIGITS FROM THE INPUT
	2009 ;	STREAM AND RETURNS THIER VALUE AS A 16 BIT BINARY
	2010 ;	INTEGER. IF MORE THAN 4 HEX DIGITS ARE ENTERED,
	2011 ;	ONLY THE LAST 4 ARE USED. THE NUMBER TERMINATES WHEN
	2012 ;	A VALID DELIMITER IS ENCOUNTERED. THE DELIMITER IS
	2013 ;	ALSO RETURNED AS AN OUTPUT OF THE FUNCTION. ILLEGAL
	2014 ;	CHARACTERS (NOT HEX DIGITS OR DELIMITER) CAUSE AN
	2015 ;	ERROR INDICATION. IF THE FIRST (VALID CHARACTER
	2016 ;	ENCOUNTERED IN THE INPUT STREAM IS NOT A DELIMITER,
	2017 ;	GETHX WILL RETURN WITH THE CARRY BIT SET TO 1;
	2018 ;	OTHERWISE, THE CARRY BIT IS SET TO 0 AND THE CONTENTS
	2019 ;	OF BC ARE UNDEFINED.
	2020 ;	
	2021 ;	GETHX:
0633 E5	2022	PUSH H ; SAVE HL
0634 210000	2023	LXI H, 0 ; INITIALIZE RESULT
0637 1E00	2024	MVI E, 0 ; INITIALIZE DIGIT FLAG TO FALSE
	2025 ;	GHX05:
0639 002006	2026	CALL GETCH ; GET A CHARACTER
063C 4F	2027	MOV C, A
063D 000506	2028	CALL ECHO ; ECHO THE CHARACTER
0640 007E07	2029	CALL VALDL ; SEE IF DELIMITER
	2030	FALSE GHX10 ; NO - BRANCH
0643 025206	2031+	JNC GHX10
0646 51	2032	MOV D, C ; YES - ALL DONE, BUT WANT TO RETURN DELIMITER
0647 E5	2033	PUSH H
0648 C1	2034	POP B ; MOVE RESULT TO BC
0649 E1	2035	POP H ; RESTORE HL
064A 7B	2036	MOV A, E ; GET FLAG
064B B7	2037	ORA A ; SET F/F'S

ORIGINAL PAGE IS
OF POOR QUALITY

IS15-11 5080/5085 MACRO ASSEMBLER, V3.0

MONTR PAGE 39

LOC	OBJ	LINE	SOURCE STATEMENT
064C	C23707	2038	JNZ SRET ; IF FLAG NON-0, A NUMBER HAS BEEN FOUND
064F	CA2396	2039	JZ FRET ; ELSE, DELIMITER WAS FIRST CHARACTER
		2040	GHX10:
0652	CD6307	2041	CALL VALDG ; IF NOT DELIMITER, SEE IF DIGIT
		2042	FALSE ERROR ; ERROR IF NOT A VALID DIGIT, EITHER
0655	D21E06	2043+	JNC ERROR
0658	CDE505	2044	CALL CHVBH ; CONVERT DIGIT TO ITS BINARY VALUE
065B	1EFF	2045	MVI E,0FFH ; SET DIGIT FLAG NON-0
065D	29	2046	DAD H ; *2
065E	29	2047	DAD H ; *4
065F	29	2048	DAD H ; *8
0660	29	2049	DAD H ; *16
0661	0600	2050	MVI B,0 ; CLEAR UPPER 8 BITS OF BC PAIR
0663	4F	2051	MOV C,A ; BINARY VALUE OF CHARACTER INTO C
0664	09	2052	DAD B ; ADD THIS VALUE TO PARTIAL RESULT
0665	C33906	2053	JMP GHX05 ; GET NEXT CHARACTER
		2054 ;	
		2055 ;	
		2056 ;	*****
		2057 ;	
		2058 ;	
		2059 ;	FUNCTION: GETNM.
		2060 ;	INPUTS: C - COUNT OF NUMBERS TO FIND IN INPUT STREAM
		2061 ;	OUTPUTS: TOP OF STACK - NUMBERS FOUND IN REVERSE ORDER (LAST ON TOP
		2062 ;	OF STACK)
		2063 ;	CALLS: GETHX,H10,ERROR
		2064 ;	DESTROYS: A,B,C,D,E,H,L,F/F'S
		2065 ;	DESCRIPTION: GETNM FINDS A SPECIFIED COUNT OF NUMBERS, BETWEEN 1
		2066 ;	AND 3, INCLUSIVE, IN THE INPUT
		2067 ;	STREAM AND RETURNS THEIR VALUES ON THE STACK. IF 2
		2068 ;	OR MORE NUMBERS ARE REQUESTED, THEN THE FIRST MUST BE
		2069 ;	LESS THAN OR EQUAL TO THE SECOND. OR THE FIRST AND
		2070 ;	SECOND NUMBERS WILL BE SET EQUAL. THE LAST NUMBER
		2071 ;	REQUESTED MUST BE TERMINATED BY A CARRIAGE RETURN
		2072 ;	OR AN ERROR INDICATION WILL RESULT.
		2073 ;	
		2074	GETNM:
0668	2E03	2075	MVI L,3 ; PUT MAXIMUM ARGUMENT COUNT INTO L
066A	79	2076	MOV A,C ; GET THE ACTUAL ARGUMENT COUNT
066B	E503	2077	ANI 3 ; FORCE TO MAXIMUM OF 3
066D	C8	2078	RZ ; IF 0, DON'T BOTHER TO DO ANYTHING
066E	67	2079	MOV H,A ; ELSE, PUT ACTUAL COUNT INTO H
		2080	GHN05:
066F	C03206	2081	CALL GETHX ; GET NUMBER FROM INPUT STREAM
		2082	FALSE ERROR ; ERROR IF NOT THERE - TOO FEW NUMBERS
0672	D21E06	2083+	JNC ERROR
0675	C5	2084	PUSH B ; ELSE, SAVE NUMBER ON STACK
0676	2D	2085	DCR L ; DECREMENT MAXIMUM ARGUMENT COUNT
0677	25	2086	DCR H ; DECREMENT ACTUAL ARGUMENT COUNT
0678	CA0406	2087	JZ GHN10 ; BRANCH IF NO MORE NUMBERS WANTED
067B	7A	2088	MOV A,D ; ELSE, GET NUMBER TERMINATOR TO A
067C	FE0D	2089	CPI CR ; SEE IF CARRIAGE RETURN
067E	CA1E05	2090	JZ ERROR ; ERROR IF SO - TOO FEW NUMBERS
0681	C36F06	2091	JMP GHN05 ; ELSE, PROCESS NEXT NUMBER
		2092	GHN10:

ORIGINAL PAGE IS
OF POOR QUALITY

ISTE-II 0000/0005 MACRO ASSEMBLER, V3.0

MONTR PAGE 40

LOC	OBJ	LINE	SOURCE STATEMENT
0684	7A	2093	MOV A,D ; WHEN COUNT 0, CHECK LAST TERMINATOR
0685	FE0D	2094	CPI CR
0687	C21E06	2095	JNZ ERROR ; ERROR IF NOT CARRIAGE RETURN
068A	01FFFF	2096	LXI B,0FFFFH ; HL GETS LARGEST NUMBER
068D	7D	2097	MOV A,L ; GET WHAT'S LEFT OF MAXIMUM ARG COUNT
068E	B7	2098	ORA A ; CHECK FOR 0
068F	CA9706	2099	JZ GNM20 ; IF YES, 3 NUMBERS WERE INPUT
		2100	GNM15:
0692	C5	2101	PUSH B ; IF NOT, FILL REMAINING ARGUMENTS WITH 0FFFFH
0693	2D	2102	DCR L
0694	C29206	2103	JNZ GNM15
		2104	GNM20:
0697	C1	2105	POP B ; GET THE 3 ARGUMENTS OUT
0698	D1	2106	POP D
0699	E1	2107	POP H
069A	CDAD06	2108	CALL HILO ; SEE IF FIRST >= SECOND
		2109	FALSE GNM25 ; NO - BRANCH
069D	02A206	2110+	JNC GNM25
06A0	54	2111	MOV D,H
06A1	5D	2112	MOV E,L ; YES - MAKE SECOND EQUAL TO THE FIRST
		2113	GNM25:
06A2	E3	2114	XTHL ; PUT FIRST ON STACK - GET RETURN ADDR
06A3	05	2115	PUSH D ; PUT SECOND ON STACK
06A4	C5	2116	PUSH B ; PUT THIRD ON STACK
06A5	E5	2117	PUSH H ; PUT RETURN ADDRESS ON STACK
		2118	GNM30:
06A6	3D	2119	DCR A ; DECREMENT RESIDUAL COUNT
06A7	F8	2120	RI ; IF NEGATIVE, PROPER RESULTS ON STACK
06A8	E1	2121	POP H ; ELSE, GET RETURN ADDR
06A9	E3	2122	XTHL ; REPLACE TOP RESULT WITH RETURN ADDR
06AA	C3A606	2123	JMP GNM30 ; TRY AGAIN
		2124 ;	
		2125 ;	
		2126 ;	*****
		2127 ;	
		2128 ;	
		2129 ;	FUNCTION: HILO
		2130 ;	INPUTS DE - 16 BIT INTEGER
		2131 ;	HL - 16 BIT INTEGER
		2132 ;	OUTPUTS: CARRY - 0 IF HL<DE
		2133 ;	- 1 IF HL>=DE
		2134 ;	CALLS: NOTHING
		2135 ;	DESTROYS: F/F'S
		2136 ;	DESCRIPTION: HILO COMPARES THE 2 16 BIT INTEGERS IN HL AND DE. THE
		2137 ;	INTEGERS ARE TREATED AS UNSIGNED NUMBERS. THE CARRY
		2138 ;	BIT IS SET ACCORDING TO THE RESULT OF THE COMPARISON.
		2139 ;	
		2140	HILO:
06AD	C5	2141	PUSH B ; SAVE BC
06AE	47	2142	MOV B,A ; SAVE A IN B REGISTER
06AF	E5	2143	PUSH H ; SAVE HL PAIR
06B0	7A	2144	MOV A,D ; CHECK FOR DE = 0000H
06B1	93	2145	ORA E
06B2	0ACE06	2146	JZ HILO5 ; WE'RE AUTOMATICALLY DONE IF IT IS
06B5	23	2147	INX H ; INCREMENT H BY 1

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V3.0

MONTR PAGE 41

LOC	OBJ	LINE	SOURCE STATEMENT
0626	7C	2148	MOV A,H ; WANT TO TEST FOR 0 RESULT AFTER
0627	25	2149	ORA L ; /INCREMENTING
0628	CACE06	2150	JZ HIL05 ; IF SO, HL MUST HAVE CONTAINED 0FFFFH
0628	E1	2151	POP H ; IF NOT, RESTORE ORIGINAL HL
062C	05	2152	PUSH D ; SAVE DE
062D	3EFF	2153	MVI A,0FFH ; WANT TO TAKE 2'S COMPLEMENT OF DE CONTENTS
062F	AA	2154	XRR D
0630	57	2155	MOV D,A
06C1	3EFF	2156	MVI A,0FFH
06C3	AB	2157	XRR E
06C4	5F	2158	MOV E,A
06C5	13	2159	INX D ; 2'S COMPLEMENT OF DE TO DE
06C6	7D	2160	MOV A,L
06C7	32	2161	ADD E ; ADD HL AND DE
06C8	7C	2162	MOV A,H
06C9	8A	2163	ADC D ; THIS OPERATION SETS CARRY PROPERLY
06CA	D1	2164	POP D ; RESTORE ORIGINAL DE CONTENTS
06CB	78	2165	MOV A,B ; RESTORE ORIGINAL CONTENTS OF A
06CC	C1	2166	POP B ; RESTORE ORIGINAL CONTENTS OF BC
06CD	C9,	2167	RET ; RETURN
		2168	HIL05:
06CE	E1	2169	POP H ; IF HL CONTAINS 0FFFFH, THEN CARRY CARRY
06CF	78	2170	MOV A,B ; /ONLY BE SET TO 1
06D0	C1	2171	POP B ; RESTORE ORIGINAL CONTENTS OF REGISTERS
06D1	C33707	2172	JMP SPET ; SET CARRY AND RETURN
		2173 ;	
		2174 ;	
		2175 ;	*****
		2176 ;	
		2177 ;	
		2178 ;	FUNCTION: NMOUT
		2179 ;	INPUTS: A - 8 BIT INTEGER
		2180 ;	OUTPUTS: NONE
		2181 ;	CALLS: ECHO,PRVAL
		2182 ;	DESTROYS: A,B,C,F/F'S
		2183 ;	DESCRIPTION: NMOUT CONVERTS THE 8 BIT, UNSIGNED INTEGER IN THE
		2184 ;	A REGISTER INTO 2 ASCII CHARACTERS. THE ASCII CHARACTERS
		2185 ;	ARE THE ONES REPRESENTING THE 8 BITS. THESE TWO
		2186 ;	CHARACTERS ARE SENT TO THE CONSOLE AT THE CURRENT PRINT
		2187 ;	POSITION OF THE CONSOLE.
		2188 ;	
		2189	NMOUT:
06D4	E5	2190	PUSH H ; SAVE HL - DESTROYED BY PRVAL
06D5	F5	2191	PUSH PSW ; SAVE ARGUMENT
06D6	0F	2192	RRC
06D7	0F	2193	RRC
06D8	0F	2194	RRC
06D9	0F	2195	RRC ; GET UPPER 4 BITS TO LOW 4 BIT POSITIONS
06DA	E50F	2196	ANI HCHAR ; MASK OUT UPPER 4 BITS - WANT 1 HEX CHAR
06DC	CD7402	2197	CALL HEXASC ; CONVERT HEX TO ASCII IN A
06DF	4F	2198	MOV C,A ; LOAD C WITH CHARACTER CODE
06E0	CD0506	2199	CALL ECHO ; SEND TO TERMINAL
06E2	F1	2200	POP PSW ; GET BACK ARGUMENT
06E4	E50F	2201	ANI HCHAR ; MASK OUT UPPER 4 BITS - WANT 1 HEX CHAR
06E6	CD7402	2202	CALL HEXASC ; CONVERT HEX TO ASCII IN A

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-11 9090/8085 MACRO ASSEMBLER, V3.0

MONTR PAGE 42

LOC	OBJ	LINE	SOURCE STATEMENT
06E9	4F	2293	MOV C,A ; LOAD C WITH CHARACTER CODE
06EA	00506	2294	CALL ECHO
06ED	51	2295	POP H ; RESTORE SAVED VALUE OF HL
06EE	09	2296	RET ; RETURN
		2297 ;	
		2298 ;	
		2299 ;	*****
		2210 ;	
		2211 ;	
		2212 ;	FUNCTION: REGDS
		2213 ;	INPUTS: NONE
		2214 ;	OUTPUTS: NONE
		2215 ;	CALLS: ECHO, NMOUT, ERROR, CROUT
		2216 ;	DESTROYS: A, B, C, D, E, H, F/F'S
		2217 ;	DESCRIPTION: REGDS DISPLAYS THE CONTENTS OF THE REGISTER SAVE
		2218 ;	LOCATIONS, IN FORMATTED FORM, ON THE CONSOLE. THE
		2219 ;	DISPLAY IS DRIVEN FROM THE TABLE RTAB, WHICH CONTAINS
		2220 ;	THE REGISTER'S PRINT SYMBOL, SAVE LOCATION ADDRESS,
		2221 ;	AND LENGTH (8 OR 16 BITS).
		2222 ;	
		2223	REGDS:
06EF	242507	2224	LXI H,RTAB ; LOAD HL WITH ADDRESS OF START OF TABLE
		2225	REGDS:
06F2	4E	2226	MOV C,M ; GET PRINT SYMBOL OF REGISTER
06F3	79	2227	MOV R,C
06F4	B7	2228	ORA A ; TEST FOR 6 - END OF TABLE
06F5	02FC06	2229	JNZ REG10 ; IF NOT END, BRANCH
06F8	00FF05	2230	CALL CROUT ; ELSE, CARRIAGE RETURN/LINE FEED TO END
06FB	09	2231	RET ; /DISPLAY
		2232	REG10:
06FC	00506	2233	CALL ECHO ; ECHO CHARACTER
06FF	0E3D	2234	MYI C,'='
0701	00506	2235	CALL ECHO ; OUTPUT EQUALS SIGN, I.E. A=
0704	23	2236	INX H ; POINT TO START OF SAVE LOCATION ADDRESS
0705	5E	2237	MOV E,H ; GET LSP OF SAVE LOCATION ADDRESS TO E
0706	1520	2238	MYI D,RAMST SHR 8 ; PUT MSP OF SAVE LOC ADDRESS INTO D
0708	23	2239	INX H ; POINT TO LENGTH FLAG
0709	1A	2240	LDAX D ; GET CONTENTS OF SAVE ADDRESS
070A	00D406	2241	CALL NMOUT ; DISPLAY ON CONSOLE
070D	7E	2242	MOV R,H ; GET LENGTH FLAG
070E	B7	2243	ORA A ; SET SIGN F/F
070F	0A1707	2244	JZ REG15 ; IF 0, REGISTER IS 8 BITS
0712	1B	2245	DCX D ; ELSE, 16 BIT REGISTER SO MORE TO DISPLAY
0713	1A	2246	LDAX D ; GET LOWER 8 BITS
0714	00D406	2247	CALL NMOUT ; DISPLAY THEM
		2248	REG15:
0717	0E20	2249	MYI C,' '
0719	00506	2250	CALL ECHO
071C	23	2251	INX H ; POINT TO START OF NEXT TABLE ENTRY
071D	03F206	2252	JMP REG05 ; DO NEXT REGISTER
		2253 ;	
		2254 ;	
		2255 ;	*****
		2256 ;	
		2257 ;	

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 9090/9085 MACRO ASSEMBLER, V3.0 MONITOR PAGE 43

LOC	OBJ	LINE	SOURCE STATEMENT
		2258	; FUNCTION: RGADR.
		2259	; INPUTS: C - CHARACTER DENOTING REGISTER
		2260	; OUTPUTS: EC - ADDRESS OF ENTRY IN RTAB CORRESPONDING TO REGISTER
		2261	; CALLS: ERROR
		2262	; DESTROYS: A, B, C, D, E, H, L, F/F'S
		2263	; DESCRIPTION: RGADR TAKES A SINGLE CHARACTER AS INPUT. THIS CHARACTER
		2264	; DENOTES A REGISTER. RGADR SEARCHES THE TABLE RTAB
		2265	; FOR A MATCH ON THE INPUT ARGUMENT. IF ONE OCCURS,
		2266	; RGADR RETURNS THE ADDRESS OF THE ADDRESS OF THE
		2267	; SAVE LOCATION CORRESPONDING TO THE REGISTER. THIS
		2268	; ADDRESS POINTS INTO RTAB. IF NO MATCH OCCURS, THEN
		2269	; THE REGISTER IDENTIFIER IS ILLEGAL AND CONTROL IS
		2270	; PASSED TO THE ERROR ROUTINE.
		2271	;
		2272	RGADR:
0720	218807	2273	LXI H, RTAB ; HL GETS ADDRESS OF TABLE START
0723	119300	2274	LXI D, RTABS ; DE GETS SIZE OF A TABLE ENTRY
		2275	RGADR:
0725	7E	2276	MOV A, M ; GET REGISTER IDENTIFIER
0727	87	2277	ORA A ; CHECK FOR TABLE END (IDENTIFIER IS 0)
0728	0A1E06	2278	JZ ERROR ; IF AT END OF TABLE, ARGUMENT IS ILLEGAL
0728	89	2279	CMP C ; ELSE, COMPARE TABLE ENTRY AND ARGUMENT
072C	0A2307	2280	JZ RGADR10 ; IF EQUAL, WE'VE FOUND WHAT WE'RE LOOKING FOR
072F	13	2281	DAD D ; ELSE, INCREMENT TABLE POINTER TO NEXT ENTRY
0730	032607	2282	JMP RGADR5 ; TRY AGAIN
		2283	RGADR10:
0733	23	2284	INX H ; IF A MATCH, INCREMENT TABLE POINTER TO
0734	44	2285	MOV B, H ; /SAVE LOCATION ADDRESS
0735	40	2286	MOV C, L ; RETURN THIS VALUE
0736	09	2287	RET ; RETURN
		2288	;
		2289	;
		2290	*****
		2291	;
		2292	;
		2293	; FUNCTION: SRET
		2294	; INPUTS: NONE
		2295	; OUTPUTS: CARRY = 1
		2296	; CALLS: NOTHING
		2297	; DESTROYS: CARRY
		2298	; DESCRIPTION: SRET IS JUMPED TO BY ROUTINES WISHING TO RETURN SUCCESS.
		2299	; SRET SETS THE CARRY TRUE AND THEN RETURNS TO THE
		2300	; CALLER OF THE ROUTINE INVOKING SRET.
		2301	;
		2302	SRET:
0737	37	2303	STC ; SET CARRY TRUE
0738	09	2304	RET ; RETURN APPROPRIATELY
		2305	;
		2306	;
		2307	*****
		2308	;
		2309	;
		2310	; FUNCTION: STHFB
		2311	; INPUTS: DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
		2312	; OUTPUTS: NONE

ORIGINAL PAGE IS
OF POOR QUALITY

IS15-11 8080/8085 MACRO ASSEMBLER, V2.0

MONTR PAGE 44

LOC	CBT	LINE	SOURCE STATEMENT
		2313	; CALLS: STHLF
		2314	; DESTROYS: A,B,C,H,L,F/F'S
		2315	; DESCRIPTION: STHF0 CHECKS THE HALF BYTE FLAG IN TEMP TO SEE IF
		2316	; IT IS SET TO LOWER. IF SO, STHF0 STORES A 0 TO
		2317	; PAD OUT THE LOWER HALF OF THE ADDRESSED BYTE;
		2318	; OTHERWISE, THE ROUTINE TAKES NO ACTION.
		2319	;
		2320	STHF0:
0739	3AFD20	2321	LDA TEMP ; GET HALF BYTE FLAG
073C	87	2322	ORA A ; SET F/F'S
073D	C0	2323	RNZ ; IF SET TO UPPER, DON'T DO ANYTHING
073E	0E00	2324	MVI C,0 ; ELSE, WANT TO STORE THE VALUE 0
0740	CD4407	2325	CALL STHLF ; DO IT
0743	C9	2326	PET ; RETURN
		2327	;
		2328	;
		2329	*****
		2330	;
		2331	;
		2332	; FUNCTION: STHLF
		2333	; INPUTS: C - 4 BIT VALUE TO BE STORED IN HALF BYTE
		2334	; DE - 16 BIT ADDRESS OF BYTE TO BE STORED INTO
		2335	; OUTPUTS: NONE
		2336	; CALLS: NOTHING
		2337	; DESTROYS: A,B,C,H,L,F/F'S
		2338	; DESCRIPTION: STHLF TAKES THE 4 BIT VALUE IN C AND STORES IT IN
		2339	; HALF OF THE BYTE ADDRESSED BY REGISTERS DE. THE
		2340	; HALF BYTE USED (EITHER UPPER OR LOWER) IS DENOTED
		2341	; BY THE VALUE OF THE FLAG IN TEMP. STHLF ASSUMES
		2342	; THAT THIS FLAG HAS BEEN PREVIOUSLY SET
		2343	; (NOMINALLY BY ICMD).
		2344	;
		2345	STHLF:
0744	05	2346	PUSH D
0745	E1	2347	POP H ; MOVE ADDRESS OF BYTE INTO HL
0746	79	2348	MOV A,C ; GET VALUE
0747	E50F	2349	ANI 0FH ; FORCE TO 4 BIT VALUE
0749	4F	2350	MOV C,A ; PUT VALUE BACK
074A	3AFD20	2351	LDA TEMP ; GET HALF BYTE FLAG
074D	87	2352	ORA A ; CHECK FOR LOWER HALF
074E	C25707	2353	JNZ STH05 ; BRANCH IF NOT
0751	7E	2354	MOV A,H ; ELSE, GET BYTE
0752	E6F0	2355	ANI 0F0H ; CLEAR LOWER 4 BITS
0754	81	2356	ORA C ; OR IN VALUE
0755	77	2357	MOV H,A ; PUT BYTE BACK
0756	C9	2358	PET ; RETURN
		2359	STH05:
0757	7E	2360	MOV A,H ; IF UPPER HALF, GET BYTE
0758	E60F	2361	ANI 0FH ; CLEAR UPPER 4 BITS
075A	47	2362	MOV B,A
075B	79	2363	MOV A,C ; GET VALUE
075C	0F	2364	RRC
075D	0F	2365	RRC
075E	0F	2366	RRC
075F	0F	2367	RRC ; ALIGN TO UPPER 4 BITS

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 8080/8085 MACRO ASSEMBLER V3.0

MONITR PAGE 45

LOC	OBJ	LINE	SOURCE STATEMENT
0750	80	2368	ORA B ; OR IN ORIGINAL 4 BITS
0751	77	2369	MOV M,A ; PUT NEW CONFIGURATION BACK
0752	09	2370	RET ; RETURN
		2371 ;	
		2372 ;	
		2373 ;	*****
		2374 ;	
		2375 ;	
		2376 ;	FUNCTION: VALDG
		2377 ;	INPUTS: C - ASCII CHARACTER
		2378 ;	OUTPUTS: CARRY - 1 IF CHARACTER REPRESENTS VALID HEX DIGIT
		2379 ;	- 0 OTHERWISE
		2380 ;	CALLS: NOTHING
		2381 ;	DESTROYS: A,F/F'S
		2382 ;	DESCRIPTION: VALDG RETURNS SUCCESS IF ITS INPUT ARGUMENT IS
		2383 ;	AN ASCII CHARACTER REPRESENTING A VALID HEX DIGIT
		2384 ;	(0-9,A-F), AND FAILURE OTHERWISE
		2385 ;	
		2386	VALDG:
0752	79	2387	MOV A,C
0754	FE00	2388	CPI '0' ; TEST IF CHARACTER IS A '0'
0755	FA2906	2389	JM FRET ; IF ASCII CODE LESS, CANNOT BE VALID DIGIT
0759	FE39	2390	CPI '9' ; ELSE, SEE IF IN RANGE '0'-'9'
076B	FA3707	2391	JM SRET ; CODE BETWEEN '0' AND '9'
076E	CA3707	2392	JZ SRET ; CODE EQUAL '9'
0771	FE41	2393	CPI 'A' ; NOT A DIGIT - TRY FOR A LETTER
0772	FA2906	2394	JM FRET ; NO - CODE BETWEEN '9' AND 'A'
0776	FE47	2395	CPI 'G'
0778	F22906	2396	JP FRET ; NO - CODE GREATER THAN 'F'
077B	C33707	2397	JMP SRET ; OKAY - CODE IS 'A' TO 'F', INCLUSIVE
		2398 ;	
		2399 ;	
		2400 ;	*****
		2401 ;	
		2402 ;	
		2403 ;	FUNCTION: VALDL
		2404 ;	INPUTS: C - CHARACTER
		2405 ;	OUTPUTS: CARRY - 1 IF INPUT ARGUMENT VALID DELIMITER
		2406 ;	- 0 OTHERWISE
		2407 ;	CALLS: NOTHING
		2408 ;	DESTROYS: A,F/F'S
		2409 ;	DESCRIPTION: VALDL RETURNS SUCCESS IF ITS INPUT ARGUMENT IS A VALID
		2410 ;	DELIMITER CHARACTER (SPACE, COMMA, CARRIAGE RETURN) AND
		2411 ;	FAILURE OTHERWISE
		2412 ;	
		2413	VALDL:
077E	79	2414	MOV A,C
077F	FE2C	2415	CPI ',' ; CHECK FOR COMMA
0781	CA3707	2416	JZ SRET
0784	FE00	2417	CPI CR ; CHECK FOR CARRIAGE RETURN
0786	CA3707	2418	JZ SRET
0789	FE20	2419	CPI ' ' ; CHECK FOR SPACE
078B	CA3707	2420	JZ SRET
078E	C32906	2421	JMP FRET ; ERROR IF NONE OF THE ABOVE
		2422 ;	

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 9080/8085 MACRO ASSEMBLER, V3.0

MONITR PAGE 46

LOC	OBJ	LINE	SOURCE STATEMENT
		2423	;
		2424	;*****
		2425	;
		2426	; DUMMY INTERRUPT ROUTINE ENTRY POINTS
		2427	;
		2428	TRAPPR:
		2429	RESET5:
		2430	RESET6:
		2431	RSET65:
		2432	RESET7:
		2433	RSET75:
0791	FB	2434	EI ; ENABLE INTERRUPTS AGAIN
0792	C9	2435	RET ; RETURN TO VECTOR POINT
		2436	;
		2437	;*****
		2438	;
		2439	;
		2440	;
		2441	;
		2442	;
		2443	;*****
		2444	;
		2445	;
		2446	SIGNON: ; SIGNON MESSAGE
0793	00	2447	DB CR,LF,'SDMON VER 1.0',CR,LF
0794	0A		
0795	52544440		
0799	4F4E2020		
079D	28564552		
07A1	20212E20		
07A5	00		
07A6	0A		
0014		2448	LSIGNON EQU \$-SIGNON ; LENGTH OF SIGNON MESSAGE
		2449	;
		2450	CADR: ; TABLE OF ADDRESSES OF COMMAND ROUTINES
07A7	0000	2451	DW 0 ; DUMMY
07A9	5005	2452	DW XCMD
07AB	3F05	2453	DW SCMD
07AD	1F05	2454	DW IACMD
07AF	0504	2455	DW ICMD
07B1	5724	2456	DW SCMD
07B2	9604	2457	DW DCMD
		2458	;
		2459	CTAB: ; TABLE OF VALID COMMAND CHARACTERS
07B5	44	2460	DB 'D'
07B6	47	2461	DB 'O'
07B7	49	2462	DB 'I'
07B8	4D	2463	DB 'M'
07B9	53	2464	DB 'S'
07BA	59	2465	DB 'X'
0006		2466	NCOMDS EQU \$-CTAB ; NUMBER OF VALID COMMANDS
		2467	;
		2468	RTAB: ; TABLE OF REGISTER INFORMATION
07BB	41	2469	DB 'R' ; REGISTER IDENTIFIER
07BC	EE	2470	DB RSRV AND OFFH ; ADDRESS OF REGISTER SAVE LOCATION

ORIGINAL PAGE IS
OF POOR QUALITY

1915-11 3030 3065 MACRO ASSEMBLER V2.0

MONITOR PAGE 47

LINE	SOURCE STATEMENT
0700 00	2471 DB 0 ; LENGTH FLAG - 0=8 BITS, 1=16 BITS
0000	2472 STABE EQU \$-STAB ; SIZE OF AN ENTRY IN THIS TABLE
0700 40	2473 DB 'B'
0700 80	2474 DB BSHV AND 0FFH
0700 C0	2475 DB 0
0701 40	2476 DB 'C'
0702 80	2477 DB CSHV AND 0FFH
0703 C0	2478 DB 0
0704 40	2479 DB 'D'
0705 80	2480 DB DSHV AND 0FFH
0706 C0	2481 DB 0
0707 40	2482 DB 'E'
0708 80	2483 DB ESHV AND 0FFH
0709 C0	2484 DB 0
070A 40	2485 DB 'F'
070B 80	2486 DB FSHV AND 0FFH
070C C0	2487 DB 0
070D 40	2488 DB 'I'
070E 80	2489 DB ISHV AND 0FFH
070F C0	2490 DB 0
0710 40	2491 DB 'H'
0711 80	2492 DB HSHV AND 0FFH
0712 C0	2493 DB 0
0713 40	2494 DB 'L'
0714 80	2495 DB LSHV AND 0FFH
0715 C0	2496 DB 0
0716 40	2497 DB 'M'
0717 80	2498 DB MSHV AND 0FFH
0718 C0	2499 DB 1
0719 40	2500 DB 'S'
071A 80	2501 DB SSVH+1 AND 0FFH
071B C0	2502 DB 1
071C 40	2503 DB 'P'
071D 80	2504 DB PSHV+1 AND 0FFH
071E C0	2505 DB 1
071F 40	2506 DB 0 ; END OF TABLE ENTRIES
0720 80	2507 DB 0
0720 C0	2508 ;
0000	2509 ORG BYTE ; BRANCH TABLE FOR USER ACCESSIBLE ROUTINES
0000 40	2510 ;
0020 02F405	2511 JMP C0 ; TTY CONSOLE OUTPUT
0023 030F05	2512 JMP C1 ; TTY CONSOLE INPUT
0023 80	2513 ;
0023 C0	2514 ;
0027	2515 TYPAD EQU 27H
0027 40	2516 ;
0027 80	2517 ;
0027 C0	2518 ;
0028 40	2519 ; CONVM/CLOCK/ FLAG1, FLAG2, FLAG3, TIMEB, TIMEB, BYTE
0028 80	2520 ;
0028 C0	2521 ; CLOCK FLAG AND COUNTERS
0029 40	2522 ;
0029 80	2523 ORG USER-5 ; 200H
0029 C0	2524 DB 0 ; 1 USEC FLAG
0030 40	2525 DB 0 ; 50 USEC FLAG

ORIGINAL PAGE IS
OF POOR QUALITY

157-1: 5054, 5065 HRCPO RESEMBLER, 47. 0

MONITOR PAGE 49

```

0000 0000 LINE SOURCE STATEMENT
2526 DB 0 ; 1 SEC FLAG
2527 DB 0 ; SECOND COUNT LSB
2528 DB 0 ; SECOND COUNT MSB
2529 DB 0 ; NOT USED
2530 ;
2531 ; *****
2532 ;
2533 ; IN THE FOLLOWING LOCATIONS, THE USER MAY PLACE JUMP INSTRUCTIONS TO
2534 ; ROUTINES FOR HANDLING THE FOLLOWING:-
2535 ; A) RST 5.6 & 7
2536 ; B) HARDWIRED USER INTERRUPT (RST 6.5)
2537 ; C) KEYBOARD "VECTORED INTERRUPT" KEY (RST 7.5)
2538 ;
2539 ORG USER ; START OF USER BRANCH LOCATIONS
2540 ;
2541 RESET5: DB 0,0,0 ; JUMP TO RST 5 ROUTINE
2542 RESET6: DB 0,0,0 ; JUMP TO RST 6 ROUTINE
2543 RESET55: DB 0,0,0 ; JUMP TO RST 6.5 (HARDWIRED USER INTERRUPT)
2544 TRAPED EQU $
2545 RESET7: DB 0,0,0 ; JUMP TO RST 7 ROUTINE
2546 USINT: DB 0,0,0 ; JUMP TO "VECTORED INTERRUPT" KEY ROUTINE
2547 ;
2548 ; *****
2549 ;
2550 SPACE IS RESERVED HERE FOR THE MONITOR STACK
2551 ;
2552 ; *****
2553 ;
2554 ORG MNSTK ; START OF MONITOR STACK
2555 ;
2556 ; SAVE LOCATIONS FOR USER REGISTERS
2557 ;
2558 ESAY: DB 0 ; E REGISTER
2559 DSAY: DB 0 ; D REGISTER
2560 CSAY: DB 0 ; C REGISTER
2561 BSAY: DB 0 ; B REGISTER
2562 FSAY: DB 0 ; FLAGS
2563 ASAY: DB 0 ; A REGISTER
2564 LSAY: DB 0 ; L REGISTER
2565 HSAY: DB 0 ; H REGISTER
2566 ISPV: DB 0 ; INTERRUPT MASK
2567 PSAY: ; PROGRAM COUNTER
2568 PCLSV: DB 0 ; LOW ORDER BYTE OF PC
2569 PCHSV: DB 0 ; HIGH ORDER BYTE OF PC
2570 SSAY: ; STACK POINTER

```

ORIGINAL PAGE IS
OF POOR QUALITY

1515-11 8000/8085 MFC80 ASSEMBLER, V3.0

MONITOR PAGE 49

LOC	CS	LINE	SOURCE STATEMENT
25F4 00		2571	SPLSV: DB 0 ; LOW ORDER BYTE OF STACK POINTER
25F5 00		2572	SPHSV: DB 0 ; HIGH ORDER BYTE OF STACK POINTER
		2573	;
		2574	;*****
		2575	;
		2576	; MONITOR STORAGE LOCATIONS
		2577	;
25F6 0000		2578	CURAD: DW 0 ; CURRENT ADDRESS
25F8 00		2579	CURDY: DB 0 ; CURRENT DATA
25F9		2580	OBUFF: DS 4 ; OUTPUT BUFFER
		2581	TEMP: ; TEMPORARY LOCATION FOR TTY MONITOR
		2582	; TEMPORARY LOCATION FOR SINGLE STEP ROUTINE
25FD 00		2583	RCPTR: DB 0 ; REGISTER POINTER
25FE 00		2584	IBUFF: DB 0 ; INPUT BUFFER
25FF 00		2585	USCSR: DB 0 ; USER SHOULD STORE IMAGE OF CSR HERE FOR EACH TIME STEP
		2586	; /CSR IS CHANGED, OTHERWISE, SINGLE STEP
		2587	; /ROUTINE WILL DESTROY CSR CONTENTS
		2588	END

PUBLIC SYMBOLS

CI	A 05CF	CNVEN	A 05EB	CO	A 05F4	DELAY	A 038B	DISP1	A 02AB	GO	A 0449	HEXASC	A 0274
MONITR	A 0000	MSEC1H	A 039D	MSEC2H	A 03A3	MSEC2H	A 0397	OUTPT	A 029E	RDKEY	A 0200	SECDT	A 0391

INTERNAL SYMBOLS

INTPT E 0000

USER SYMBOLS

ADCPW	A 02F8	ADFLD	A 0000	ADISP	A 0090	ASAV	A 20EE	BLANK	A 00A0	BLANKS	A 03DB	BLNKS	A 030F
BACMR	A 001B	BRTAB	A 0320	BSAV	A 20EC	CBOR	A 07A7	CDATA	A 000C	CI	A 05CF	CLCER	A 0005
CLOIS	A 0106	CLOST	A 010C	CLEAR	A 010D	CM010	A 0087	CM015	A 0093	CM0AD	A 039A	CM0TB	A 03B5
CMAND	A 0024	CNIN	A 000C	CNOUT	A 000C	CNTRL	A 0001	CNVEN	A 05EB	CO	A 05F4	COLCNT	A 02E2
CONMA	A 0011	CONCP	A 000D	CONSCP	A 000E	CR	A 0000	CROUT	A 05FF	CSAV	A 20EB	CSNIT	A 0000
CSR	A 0020	CSTAT	A 000D	CTAB	A 07E5	CURAD	A 20F6	CURDT	A 20F8	DCM05	A 049D	DCM10	A 0498
DCM15	A 049D	DCMD	A 0486	DDISP	A 0094	DELAY	A 038B	DELENT	A 03AA	DISP1	A 02AB	DISPC	A 01EB
DOT	A 0001	DSAV	A 20EA	DSFDT2	A 0390	DSFDT4	A 0373	DSPLY	A 00D0	DSPLNUM	A 03C4	DSPTB	A 03C4
EFLD	A 0001	ECH05	A 058E	ECH10	A 061C	ECH0	A 0605	EIGHT	A 00F8	EMPTY	A 0000	ENC00E	A 02F3
ERM05	A 03E3	ERR	A 01FF	ERROR	A 061E	ESAV	A 20E9	ESC	A 001B	EXAM	A 009E	EXIT	A 0623
EXM05	A 00AF	EXM10	A 00C8	EXM05	A 03E7	FALSE	+ 0001	FIVE	A 0005	FRET	A 0403	FSAY	A 20ED
G10	A 00FA	GCM05	A 040C	GCM10	A 0402	GCM0	A 0487	GETCH	A 062C	GETCH	A 0457	GETHX	A 0633
GETHM	A 0668	GHX05	A 0639	GHX10	A 0632	GXM05	A 066F	GXM10	A 0624	GXM15	A 0692	GXM20	A 0697
GXM25	A 06A2	GXM30	A 06A6	GO	A 0449	GUCMD	A 000B	GTC03	A 0462	GTC05	A 0470	GTC10	A 047C
GTH05	A 0214	GTH10	A 0231	GTH20	A 0237	GTH25	A 0248	GTH05	A 0200	HCHAR	A 000F	HEX05	A 027E
HEXASC	A 0274	HIL05	A 060E	HIL0	A 06A0	HSAY	A 20F0	HXDSP	A 024E	HXDSP2	A 0260	IBUFF	A 20FE
ICH05	A 04E0	ICH10	A 0508	ICH20	A 0510	ICH25	A 0516	ICMD	A 04D5	ININT	A 0281	INITUT	A 0000
INSDG	A 0286	INTPT	E 0000	INVRT	A 00FF	ISAV	A 20F1	LETRA	A 00C1	LETRB	A 00C2	LETRC	A 00C3
LETRD	A 00C4	LETR	A 00C5	LETRF	A 00C6	LETRH	A 00C8	LETRI	A 00C9	LETRL	A 00CC	LETRP	A 0000
LETRR	A 00D2	LETRS	A 00D3	LF	A 000A	LOHER	A 0000	LSAY	A 20EF	LSGNON	A 0014	MCM05	A 0527
MCM0	A 051F	MONITR	A 0000	MONITR	A 0000	MONPRK	A 007E	MSEC1H	A 039D	MSEC20	A 03A3	MSEC2H	A 0397
MSECDT	A 03A6	MSGL	A 044E	MCMDS	A 0005	NEHLN	A 000F	NMOUT	A 00D4	INITEL	A 0009	NOCOT	A 0000
NKEY	A 0074	NUMC	A 0005	NUMRG	A 0000	NXTCOL	A 02C4	NXTRG	A 029F	OBUFF	A 20F9	OUT05	A 02A0
OUTPT	A 029E	PCHSV	A 20F3	PELEV	A 20F2	PERIO	A 0010	PEMPT	A 00F0	PEV0	A 0075	FSAY	A 20F2
RAMST	A 2000	REF	A 0040	RDKEY	A 02C0	RDYTEL	A 03F3	REG05	A 06F2	REG10	A 06FC	REG15	A 0717
REGDS	A 05EF	RES10	A 003F	RESET5	A 0791	RESET6	A 0791	RESET7	A 0791	SETF	A 0103	SETT	A 030B
RGA05	A 0726	RGA10	A 0732	RGA0R	A 0720	RGL0C	A 0300	RGNAM	A 0314	RCPTB	A 03F0	RCPTR	A 20FD

ORIGINAL PAGE IS
OF POOR QUALITY

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0

MONTR PAGE 50

RGTEL A 0430	RKEXIT A 02FE	RMUSE A 0017	RONCNT A 02EA	REET5 A 2003	REET6 A 2008	REET65 A 0791
RSET7 A 2001	RSET75 A 0791	REP05 A 0310	PER10 A 0341	PST65 A 20CE	PST05 A 0328	RTAB A 0788
RTAB5 A 0003	RTAB5 A 0544	SCM10 A 054F	SCM15 A 055F	SCMD A 052F	SECOT A 0291	SETR6 A 0354
SCMCH A 0793	SKLN A 0012	SHSG A 03EF	SPHSV A 20F5	SPLSV A 20F4	SPET A 0737	SHV A 20F4
SSSTEP A 0108	STH05 A 0757	STHF0 A 0739	STHLF A 0744	STP20 A 0120	STP21 A 0142	STP22 A 0149
STP23 A 014C	STP25 A 014F	SUB05 A 0194	SUB10 A 019A	SUB15 A 01C5	SUBST A 017F	T6E A 0080
TEMP A 20FD	TERM A 001B	TRAFEP A 20D1	TRAFPP A 0791	TRUE + 0000	TTYADV A 0027	UERLN A 000F
UNMSK A 000E	UPD0D A 036F	UPDOT A 037C	UPPER A 00FF	URTPC A 00E0	URTSPC A 0003	USCSR A 20FF
USTNT A 20D4	USRBR A 20C8	VALDG A 0763	VALDL A 077E	XCM05 A 0576	XCM10 A 0585	XCM15 A 0592
XCM18 A 059D	XCM20 A 05B6	XCM25 A 05CD	XCM27 A 05CE	XCM30 A 05D6	XCM0 A 0562	YMS0 A 03EB
ZERO A 00B0						

ASSEMBLY COMPLETE, NO ERRORS

SUBMIT FORTEN(VCSFPG)
SUBMIT FORTEN(VCSSEP)
SUBMIT FORTEN(VCSSEH)
SUBMIT FORTEN(VCSCHT)
SUBMIT FORTEN(VCSDES)
SUBMIT FORTEN(VCSIOU)
SUBMIT FORTEN(VCSCON)
SUBMIT FORTEN(VCSINP)
SUBMIT FORTEN(VCSIOZ)
SUBMIT FORTEN(VCSAPP)
SUBMIT ASMBLP(VCSUTL)
SUBMIT ASMBLP(VCSIOX)
SUBMIT ASMBLP(VCSIOD)
SUBMIT ASMBLP(INTRPT)
SUBMIT ASMBLP(DEFAULT)
SUBMIT ASMBLP(TAELES)
SUBMIT ASMBLP(DEFTAB)
SUBMIT ASMBLP(MONITR)

ORIGINAL PAGE IS
OF POOR QUALITY

REPORTS FILE NO FOR
COPY FILE NO LIST TO FILE
DELETE FILE NO LIST

ORIGINAL PAGE IS
OF POOR QUALITY

ASAC - F1 NO ASM MICROFILE INDEX
CORP - F1 NO LST TO ALP
DELETE - F1 NO LST

LINK -F1-WCSPG OBJ -F1-WCSEP OBJ, &
-F1-WCEXS LIB -F1-WCSEP OBJ, &
-F1-MONTAB LIB, &
-F1-WCSTAS LIB -F1-WCSTOL LIB, &
-F1-TABLES OBJ -F1-DEFTAB OBJ -F1-DEFLT OBJ, &
FROPUN LIB, FROHIO LIB, FREE LIB, &
FRHILL LIB, FRMGA LIB, &
TO -F1-WCSPG1 LNK

ORIGINAL PAGE IS
OF POOR QUALITY

```

DELETE :F1:VCS*.LIB, :F1:DEBUG.LIB, :F1:MONTRB.LIB
LIB
CREATE :F1:VCSSEPS.LIB
ADD :F1:VCSSEUP.OBJ TO :F1:VCSSEPS.LIB
CREATE :F1:VCSICS.LIB
ADD :F1:VCSINP.OBJ TO :F1:VCSICS.LIB
ADD :F1:VCSOON.OBJ TO :F1:VCSICS.LIB
CREATE :F1:MONTRB.LIB
ADD :F1:MONTRB.OBJ TO :F1:MONTRB.LIB
ADD :F1:INTPTB.OBJ TO :F1:MONTRB.LIB
ADD :F1:TABLES.OBJ TO :F1:MONTRB.LIB
ADD :F1:DEFLT.OBJ TO :F1:MONTRB.LIB
ADD :F1:TESTB.OBJ TO :F1:MONTRB.LIB
CREATE :F1:VCSXS.LIB
ADD :F1:VCSCHT.OBJ TO :F1:VCSXS.LIB
ADD :F1:VCSXN1.OBJ TO :F1:VCSXS.LIB
ADD :F1:VCSXAP.OBJ TO :F1:VCSXS.LIB
CREATE :F1:DEBUG.LIB
ADD :F1:VCSDBG.OBJ TO :F1:DEBUG.LIB
CREATE :F1:VCSICL.LIB
ADD :F1:VCSIOU.OBJ TO :F1:VCSICL.LIB
ADD :F1:VCSUTL.OBJ TO :F1:VCSICL.LIB
ADD :F1:VCSIOO.OBJ TO :F1:VCSICL.LIB
ADD :F1:VCSIOZ.OBJ TO :F1:VCSICL.LIB
ADD :F1:VCSIOX.OBJ TO :F1:VCSICL.LIB
EXIT

```

ORIGINAL PAGE IS
OF POOR QUALITY

DELETE FI-WSIOL LIS
LIE
CREATE FI-WSIOL LIS
ADD FI-WSIOL 001 TO FI-WSIOL LIS
ADD FI-WSIOL 001 TO FI-WSIOL LIS
ADD FI-WSIOL 001 TO FI-WSIOL LIS
ADD FI-WSIOL 001 TO FI-WSIOL LIS
ADD FI-WSIOL 001 TO FI-WSIOL LIS
ADD FI-WSIOL 001 TO FI-WSIOL LIS
EXIT

ORIGINAL PAGE IS
OF POOR QUALITY